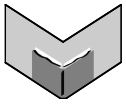
The Journal of Machine Learning Research
Volume 14
Print-Archive Edition

Pages 2519–3818



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Volume 14
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 2013.

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 14, 2013

Journal of Machine Learning Research 14 (2013) 2519-2548 Submitted 10/12; Revised 5/13; Published 9/13

Learning Bilinear Model for Matching Queries and Documents

Wei Wu WUWEI@MICROSOFT.COM
Microsoft Research Asia
13F, Building 2
No.5 Danling Street
Beijing, 100080, P. R. China

Zhengdong Lu LU.ZHENGDONG@HUAWEI.COM
Hang Li HANGLI.HL@HUAWEI.COM
Huawei Noah's Ark Lab
Units 525-530, Core Building 2
Hong Kong Science Park
Shatin, Hong Kong

Editor: Charles Elkan

Abstract

The task of matching data from two heterogeneous domains naturally arises in various areas such as web search, collaborative filtering, and drug design. In web search, existing work has designed relevance models to match queries and documents by exploiting either user clicks or content of queries and documents. To the best of our knowledge, however, there has been little work on principled approaches to leveraging both clicks and content to learn a matching model for search. In this paper, we propose a framework for learning to match heterogeneous objects. The framework learns two linear mappings for two objects respectively, and matches them via the dot product of their images after mapping. Moreover, when different regularizations are enforced, the framework renders a rich family of matching models. With orthonormal constraints on mapping functions, the framework subsumes Partial Least Squares (PLS) as a special case. Alternatively, with a ℓ_1+ℓ_2 regularization, we obtain a new model called *Regularized Mapping to Latent Structures* (RMLS). RMLS enjoys many advantages over PLS, including lower time complexity and easy parallelization. To further understand the matching framework, we conduct generalization analysis and apply the result to both PLS and RMLS. We apply the framework to web search and implement both PLS and RMLS using a click-through bipartite with metadata representing features of queries and documents. We test the efficacy and scalability of RMLS and PLS on large scale web search problems. The results show that both PLS and RMLS can significantly outperform baseline methods, while RMLS substantially speeds up the learning process.

Keywords: web search, partial least squares, regularized mapping to latent structures, generalization analysis

1. Introduction

Many tasks in machine learning and data mining can be formalized as matching between objects from two spaces. One particular example is web search, where the retrieved documents are ordered according to their relevance to the given query. The relevance is determined by the matching scores between the query and the documents. It is therefore crucial to accurately calculate the matching

score for any given query-document pair. Similarly, matching between heterogeneous data sources can be found in collaborative filtering, image annotation, drug design, etc.

Existing models in web search use information from different sources to match queries and documents. On one hand, conventional relevance models, including Vector Space Model (VSM) (Salton and McGill, 1986), BM25 (Robertson et al., 1994), and Language Models for Information Retrieval (LMIR) (Ponte and Croft, 1998; Zhai and Lafferty, 2004), match queries and documents based on their content. Specifically, queries and documents are represented as feature vectors in a Euclidean space, and conventional relevance models match them by the dot products of their feature vectors (Xu et al., 2010; Wu et al., 2011). On the other hand, a click-through bipartite graph, which represents users' implicit judgments on query-document relevance, has proven to be a very valuable resource for matching queries and documents. Many methods have been proposed (Craswell and Szummer, 2007; Ma et al., 2008), but they rely only on the structure of the bipartite graph. Existing models rely on either features or the click-through bipartite graph to match queries and documents. Therefore, there is need for a principled approach to learning to match with both features and the click-through bipartite graph. The learnt model must maintain efficacy and scalability (due to the massive scale of web search problems). Moreover, we also would like to understand the generalization ability of matching models.

This paper proposes a general framework for learning to match objects from two spaces. Specifically, the framework learns a linear mapping for each object and map the two objects into a common latent space. After that, the dot product of the images of the two objects is taken as their matching score. The matching model is linear in terms of both objects, and therefore, we actually learn a bilinear model for matching two objects. The types of linear mapping can be further specified by a set of constraints. By limiting the mappings to projections,[1] we obtain a natural generalization to Partial Least Squares (PLS), a classic model in statistics for analyzing the correlation between two variables. More interestingly, when replacing this constraint with regularization constraints based on ℓ_1 and ℓ_2 norms, we get a new learning to match model, called Regularized Mapping to Latent Structures (RMLS). This model allows easy parallelization for learning and fast computation in testing due to the induced sparsity in the mapping matrices. More specifically, RMLS allows pre-computing intermediate parameters, making optimization independent of training instances. Moreover, the pre-computation can be easily distributed across different machines, and therefore can further enhance the efficiency and scalability of RMLS. To further understand this framework, we give a generalization analysis under a hierarchical sampling assumption which is natural in the matching problems encountered in web search. Our results indicate that to obtain a good generalization ability, it is necessary to use a large number of instances for each type of object.

With the framework, we learn a matching model for search by leveraging both click-through and features. Specifically, we implement both PLS and RMLS using a click-through bipartite graph with metadata on the nodes representing features of queries and documents. We take click numbers as a response and learn linear mappings to matching queries and documents, each represented by heterogeneous feature vectors consisting of both key words and click numbers. On a small data set, RMLS and PLS perform comparably well and both of them significantly outperform other baseline methods. RMLS is more efficient than PLS and the advantage becomes more significant under parallelization. RMLS also scales well on large data sets. On a data set with millions of queries and

1. In this paper, by projection, we mean a linear mapping specified by a matrix P, with $P^\top P = \mathbb{I}$.

documents, RMLS can leverage high dimensional features and significantly outperform all other baseline methods.

Our contributions are three-fold: 1) we propose a framework for learning to match heterogeneous data, with PLS and the more scalable RMLS as special cases; 2) generalization analysis of this framework as well as its application to RMLS and PLS; 3) empirical verification of the efficacy and scalability of RMLS and PLS on real-world large-scale web search data.

2. Related Work

Matching pairs of objects with a similarity function defined as a dot product has been researched for quite some time. When the pair of objects are from the same space (i.e., they are homogeneous data), the similarity function becomes positive semi-definite, and the matching problem is essentially finding a good kernel (Cristianini et al., 2001; Lanckriet et al., 2002; Bach et al., 2004; Ong et al., 2005; Micchelli and Pontil, 2005; Bach, 2008; Varma and Babu, 2009; Cortes, 2009). Among existing works, distance metric learning (Jolliffe, 2002; Xing et al., 2003; Schultz and Joachims, 2003; JacobGoldberger and GeoffHinton, 2004; Hertz et al., 2004; Hoi et al., 2006; Yang et al., 2006; Davis et al., 2007; Sugiyama, 2007; Weinberger and Saul, 2009; Ying et al., 2009) is a representative approach of learning similarities (or dissimilarities) for homogeneous data. In distance metric learning, a linear transformation is learnt for mapping objects from the same space into a latent space. In the space, dot product or Euclidean distance is taken as a means to measure similarity or dissimilarity. Recently, learning a similarity function for object pairs from two different spaces has also emerged as a hot research topic (Grangier and Bengio, 2008; Abernethy et al., 2009). Our model belongs to the latter category, but is tailored for web search and tries to solve problems central to that, for example, scalability.

Our model, when applied to web search, is also obviously related to the effort on learning to rank (Herbrich et al., 1999; Crammer and Singer, 2001; Joachims, 2002; Agarwal and Niyogi, 2005; Rudin et al., 2005; Burges et al., 2006; Cao et al., 2006; Xu and Li, 2007; Cao et al., 2007; Liu, 2009; Li, 2011). However, we focus on learning to match queries and documents, while learning to rank has been more concerned with optimizing the ranking model. Clearly the matching score learned with our method can be integrated as a feature for a particular learning to rank model, and therefore our model is in a sense feature learning for learning to rank.

In web search, existing work for matching queries and documents can be roughly categorized into two groups: feature based methods and graph based methods. In the former group, Vector Space Model (VSM) (Salton and McGill, 1986), BM25 (Robertson et al., 1994), and Language Models for Information Retrieval (LMIR) (Ponte and Croft, 1998; Zhai and Lafferty, 2004) make use of features, particularly, n-gram features to calculate query-document matching scores. As pointed out by Xu et al. (2010) as well as Wu et al. (2011), these models perform matching by using the dot product between a query vector and a document vector as a query-document similarity function. In the latter group, graph based methods exploit the structure of a click-through bipartite graph to match query-document pairs. For example, Latent Semantic Indexing (LSI) (Deerwester et al., 1990) can be employed, which uses SVD to project queries and documents in a click-through bipartite graph into a latent space, and calculates query-document matching scores through the dot product of their images in the latent space. Craswell and Szummer (2007) propose adopting a backward random walk process on a click-through bipartite graph to propagate similarity through probabilistic transitions. In this paper, we propose a general framework for matching queries and

documents. The framework can leverage both features and a click-through bipartite graph to learn a matching model. In doing so, we actually combine feature based methods and graph based methods in a principled way.

In information retrieval, some recent work also considers leveraging both user clicks and content of queries and documents. Bai et al. (2009) propose learning a low rank model for ranking documents, which is like matching queries and documents. On the other hand, there are also stark differences between our work and theirs. For example, their work requires a pair-wise input supervision and learns a ranking model using hinge loss, while our work employs a point-wise input and learns a matching model using alignment. Gao et al. (2011) propose combining ideas in semantic representation and statistical machine translation to learn relevance models for web search. Compared with their work, our method is non-probabilistic and can leverage different regularizations, such as the ℓ_1 regularization, to achieve better performance.

The matching problem is also widely studied in collaborative filtering (CF) whose goal can be viewed as matching users and items (Hofmann, 2004; Srebro et al., 2005; Abernethy et al., 2009; Chen et al., 2012). The characteristics of the problems CF attempts to solve, for example, the sampling assumption and the nature of "ratings", are different from the matching problem in web search. We have compared our methods with a state-of-the-art CF model which can handle extra attributes in two domains. The results indicate that our methods are more effective in web search than the existing CF model.

In existing statistical models, Partial Least Squares (PLS) (Rosipal and Krämer, 2006; Schreier, 2008) and Canonical Correlation Analysis (CCA) (Hardoon et al., 2004) are classic tools for capturing correlations of two variables via common latent structures. In this paper, we provide a general matching framework, which subsumes PLS and allows rather scalable implementations.

3. A Framework For Matching Objects From Two Spaces

We first give a general framework for learning to match objects from two heterogeneous spaces. Suppose that there are two spaces $\mathcal{X} \subset \mathbb{R}^{d_x}$ and $\mathcal{Y} \subset \mathbb{R}^{d_y}$. For any $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, there is a response $r \doteq r(x,y) \geqslant 0$ in space $\mathcal{R}$, indicating the actual correlation between object x and object y. For web search, the objects are queries and documents, and the response can be judgment from human labelers or the click number from user logs.

We first describe the hierarchical sampling process for generating any sample triple (x_i, y_{ij}, r_{ij}).

Assumption 1 *First x_i is sampled according to $P(x)$. Then y_{ij} is sampled according to $P(y|x_i)$. After that, there is a response $r_{ij} = r(x_i, y_{ij})$ associated with pair (x_i, y_{ij}).*

We argue that this is an appropriate sampling assumption for web search (Chen et al., 2010), since the selected y_{ij} (in this case, retrieved document) depends heavily on x_i (in this case, query). This dependence is largely rendered by several factors of a search engine, including the indexed pages and the ranking algorithms. Under Assumption 1, we have a sample set $\mathcal{S} = \{(x_i, y_{ij}, r_{ij})\}$, with $1 \leqslant i \leqslant n^x$, and for any given i, $1 \leqslant j \leqslant n_i^y$. Here $\{x_i\}_{i=1}^{n^x}$ are i.i.d. sampled and for a given x_i, $\{y_{ij}\}_{j=1}^{n_i^y}$ are i.i.d. samples conditioned on x_i. Relying on this sampling assumption, we will give the learning to match framework, and later carry out the generalization analysis.

3.1 Model

We intend to find a linear mapping pair (L_x, L_y), so that the corresponding images $L_x^\top x$ and $L_y^\top y$ are in the same d-dimensional latent space $\mathcal{L}$ (with $d \ll \min\{d_x, d_y\}$), and the degree of matching between x and y can be reduced to the dot product in $\mathcal{L}$:

$$\mathsf{match}_{L_x,L_y}(x,y) = x^\top L_x L_y^\top y.$$

Dot product is a popular form of matching in applications like search. In fact, traditional relevance models in search such as VSM (Salton and McGill, 1986), BM25 (Robertson et al., 1994), and LMIR (Ponte and Croft, 1998; Zhai and Lafferty, 2004) are all dot products of a query vector and a document vector, as pointed out by Xu et al. (2010) as well as Wu et al. (2011). Recently, Bai et al. (2009) proposed supervised semantic indexing, which also uses dot product as the measure of query-document similarity. We hope the score defined this way can reflect the actual response. More specifically, we would like to maximize the following expected alignment between this matching score and the response

$$\mathbb{E}_{x,y}\{r(x,y) \cdot \mathsf{match}_{L_x,L_y}(x,y)\} = \mathbb{E}_x \mathbb{E}_{y|x}\{r(x,y) x^\top L_x L_y^\top y\}, \tag{1}$$

which is in the same spirit as the technique used by Cristianini et al. (2001) for kernel learning. The expectation in (1) can be estimated as

$$\frac{1}{n^x} \sum_{i=1}^{n^x} \frac{1}{n_i^y} \sum_{j=1}^{n_i^y} r_{ij} x_i^\top L_x L_y^\top y_{ij}.$$

The learning problem hence boils down to

$$\begin{aligned} \underset{L_x,L_y}{\arg\max} \quad & \frac{1}{n^x} \sum_{i=1}^{n^x} \frac{1}{n_i^y} \sum_{j=1}^{n_i^y} r_{ij} x_i^\top L_x L_y^\top y_{ij}, \\ \text{s.t.} \quad & L_x \in \mathcal{H}_x,\ L_y \in \mathcal{H}_y, \end{aligned} \tag{2}$$

where $\mathcal{H}_x$ and $\mathcal{H}_y$ are hypothesis spaces for L_x and L_y respectively. Since the final matching model is linear in terms of both x and y, Framework (2) actually learns a bilinear model for matching objects from two spaces.

3.2 Special Cases

The matching framework in (2) defines a rather rich family of matching models, with different choices of $\mathcal{H}_x$ and $\mathcal{H}_y$. More specifically, we define $\mathcal{H}_x = \{L_x \mid L_x^\top L_x = \mathbb{I}_{d\times d}\}$ and $\mathcal{H}_y = \{L_y \mid L_y^\top L_y = \mathbb{I}_{d\times d}\}$. In other words, both L_x and L_y are confined to be matrices with orthonormal columns, then the program in (2) becomes a natural extension to Partial Least Squares (PLS) (Rosipal and Krämer, 2006; Schreier, 2008). Like the well-known Canonical Correlation Analysis (CCA) (Hardoon et al., 2004), PLS is also a classic statistical model for capturing the correlation between two variables. In contrast, in (2) we allow an instance in one domain to be associated with multiple instances in the other domain, and argument each association with a weight (response). This extension enables us to model the complex bipartite association relations in matching tasks. To see the connection between

(1) and traditional PLS, we re-write (2) as

$$\begin{aligned} \underset{L_x, L_y}{\arg\max} \quad & \frac{1}{n^x}\sum_{i=1}^{n^x} x_i^\top L_x L_y^\top y_i' = \mathrm{trace}(L_y^\top (\frac{1}{n^x}\sum_{i=1}^{n^x} y_i' x_i^\top) L_x), \\ \text{s.t.} \quad & L_x^\top L_x = L_y^\top L_y = \mathbb{I}_{d\times d}, \end{aligned}$$

where $y_i' = 1/n_i^y \sum_{j=1}^{n_i^y} r_{ij} y_{ij}$. The program is exactly the formulation of PLS as formulated by Schreier (2008) when viewing y_i' a variable.[2]

Interestingly, our framework in (2) also subsumes the Latent Semantic Index (LSI) (Deerwester et al., 1990) used in information retrieval. Specifically, suppose that $\mathcal{X}$ represents document space and $\mathcal{Y}$ represents term space. Response r represents the tf-idf weight of a term y in a document x. Let x and y be the indicator vectors of a document and a term, that is, there is only non-zero element *one* in x and y at the location indexing the corresponding document or term. The objective function in (2) becomes trace $\left(L_y^\top (\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y} r_{ij} y_{ij} x_i^\top) L_x\right)$ after ignoring n^x and n_i^y, which is exactly the objective for the SVD in LSI assuming the same orthonormal $\mathcal{H}_x$ and $\mathcal{H}_y$ defined for PLS.

The orthonormal constraints in PLS requires SVD of large matrices (Schreier, 2008), rendering it impractical for web scale applications (e.g., millions of objects with millions of features in basic settings). In next section we will consider other choices of $\mathcal{H}_x$ and $\mathcal{H}_y$ for more scalable alternatives.

4. Regularized Mapping to Latent Structures

Heading towards a more scalable matching model, we drop the orthonormal constraints in PLS, and replace them with ℓ_1 norm and ℓ_2 norm based constraints on L_x and L_y. More specifically, we define the following hypothesis spaces

$$\begin{aligned} \mathcal{H}_x &= \{L_x \mid |l_{xu}| \leqslant \lambda_x, \|l_{xu}\| \leqslant \theta_x, u = 1, \ldots, d_x\}, \\ \mathcal{H}_y &= \{L_y \mid |l_{yv}| \leqslant \lambda_y, \|l_{yv}\| \leqslant \theta_y, v = 1, \ldots, d_y\}, \end{aligned}$$

where $|\cdot|$ and $\|\cdot\|$ are respectively the ℓ_1-norm and ℓ_2-norm, l_{xu} and l_{yv} are respectively the u^{th} and v^{th} row of L_x and L_y, $\{\lambda_x, \theta_x, \lambda_y, \theta_y\}$ are parameters. Here the ℓ_1-norm based constraints will induce row-wise sparsity in L_x and L_y. The ℓ_2-norm on rows, in addition to posing further regularization, avoids degenerative solutions (see Appendix A for details). The row-wise sparsity in L_x and L_y in turn yields sparse images in $\mathcal{L}$ with sparse x and y. Indeed, for any $x = [x^{(1)} \ \ldots \ x^{(d_x)}]^\top$, its image in $\mathcal{L}$ is $L_x^\top x = \sum_{u=1}^{d_x} x^{(u)} l_{xu}$. When both x and l_{xu} are sparse, $L_x^\top x$ is the sum of a few sparse vectors, and therefore likely to be sparse itself. A similar scenario holds true for y. In web search, it is usually the case that both x and y are extremely sparse. Sparse mapping matrices and sparse images in latent structures will mitigate the memory pressure and enhance efficiency in both training and testing. With $\mathcal{H}_x$ and $\mathcal{H}_y$ defined above, we have the following program:

$$\begin{aligned} \underset{L_x, L_y}{\arg\max} \quad & \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} r_{ij} x_i^\top L_x L_y^\top y_{ij}, \\ s.t. \quad & |l_{xu}| \leqslant \lambda_x, \ \|l_{xu}\| \leqslant \theta_x, \ |l_{yv}| \leqslant \lambda_y, \ \|l_{yv}\| \leqslant \theta_y, \quad 1 \leqslant u \leqslant d_x, \ 1 \leqslant v \leqslant d_y. \end{aligned} \tag{3}$$

The matching model defined in (3) is called *Regularized Mapping to Latent Structures* (RMLS).

2. We can assume that x and y' are centered.

4.1 Optimization

In practice, we instead solve the following penalized variant of (3) for easier optimization

$$\begin{aligned} \underset{L_x,L_y}{\arg\min} \quad & -\frac{1}{n^x}\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n_i^y}r_{ij}x_i^\top L_x L_y^\top y_{ij} + \beta\sum_{u=1}^{d_x}|l_{xu}| + \gamma\sum_{v=1}^{d_y}|l_{yv}|, \\ s.t. \quad & \|l_{xu}\| \leqslant \theta_x, \|l_{yv}\| \leqslant \theta_y,\ 1 \leqslant u \leqslant d_x, 1 \leqslant v \leqslant d_y, \end{aligned} \tag{4}$$

where $\beta > 0$ and $\gamma > 0$ control the trade-off between the objective and the penalty. We employ the coordinate descent technique to solve problem (4). Since the objective in (4) is not convex, there is no guarantee for convergence to a global minimum.

Specifically, for a fixed L_y, the objective function of problem (4) can be re-written as

$$\sum_{u=1}^{d_x}\left(-(\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n^x n_i^y}x_i^{(u)}r_{ij}L_y^\top y_{ij})^\top l_{xu} + \beta|l_{xu}|\right).$$

Representing the d-dimensional $\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n^x n_i^y}x_i^{(u)}r_{ij}L_y^\top y_{ij}$ as $\omega_u = [\omega_u^{(1)}, \omega_u^{(2)}, \ldots, \omega_u^{(d)}]^\top$, the optimal l_{xu} is given by

$$l_{xu}^{(z)*} = C_u \cdot \left(\max(|\omega_u^{(z)}| - \beta, 0)\text{sign}(\omega_u^{(z)})\right),\ 1 \leqslant z \leqslant d, \tag{5}$$

where $l_{xu}^{(z)}$ represents the z^{th} element of l_{xu}. $\text{sign}(\cdot)$ represents the sign function. C_u is a constant that makes $\|l_{xu}^*\| = \theta_x$ if there are nonzero elements in l_{xu}^*, otherwise $C_u = 0$.

Similarly, for a fixed L_x, the objective function of problem (4) can be re-written as

$$\sum_{v=1}^{d_y}\left(-(\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n^x n_i^y}y_{ij}^{(v)}r_{ij}L_x^\top x_i)^\top l_{yv} + \gamma|l_{yv}|\right).$$

Writing $\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n^x n_i^y}y_{ij}^{(v)}r_{ij}L_x^\top x_i$ as $\eta_v = [\eta_v^{(1)}, \ldots, \eta_v^{(d)}]^\top$, the optimal l_{yv} is given by

$$l_{yv}^{(z)*} = C_v \cdot \left(\max(|\eta_v^{(z)}| - \gamma, 0)\text{sign}(\eta_v^{(z)})\right),\ 1 \leqslant z \leqslant d, \tag{6}$$

where $l_{yv}^{(z)}$ represents the z^{th} element of l_{yv}. C_v is a constant that makes $||l_{yv}^*|| = \theta_y$ if there are nonzero elements in l_{yv}^*, otherwise $C_v = 0$. Note that $\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n^x n_i^y}x_i^{(u)}r_{ij}L_y^\top y_{ij} = L_y^\top w_{xu}$, where $w_{xu} = \sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n^x n_i^y}x^{(u)}r_{ij}y_{ij}$ does not rely on the update of L_x and L_y and can be pre-calculated to save time. Similarly we pre-calculate $w_{yv} = \sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{1}{n^x n_i^y}y_{ij}^{(v)}r_{ij}x_i$.

The preprocessing is described in Algorithm 1, whose time complexity is $O(d_x N_x \tilde{n}^y c_y + d_y N_y \tilde{n}^x c_x)$, where N_x stands for the average number of nonzeros in all x samples per dimension, N_y is the average number of nonzeros in all y samples per dimension, $\tilde{n}^x$ is the average number of related x samples per y, $\tilde{n}^y$ is the mean of n_i^y, c_x is the average number of nonzeros in each x sample, and c_y is the average number of nonzeros in each y sample.

Algorithm 1 Preprocessing

Input: $\mathcal{S} = \{(x_i, y_{ij}, r_{ij})\}$, $1 \leqslant i \leqslant n^x$, and $1 \leqslant j \leqslant n_i^y$.
for $u = 1 : d_x$
  $w_{xu} \leftarrow \mathbf{0}$
for $v = 1 : d_y$
  $w_{yv} \leftarrow \mathbf{0}$
for $u = 1 : d_x$, $i = 1 : n^x$, $j = 1 : n_i^y$
  $w_{xu} \leftarrow w_{xu} + \frac{1}{n^x n_i^y} x_i^{(u)} r_{ij} y_{ij}$
for $v = 1 : d_y$, $i = 1 : n^x$, $j = 1 : n_i^y$
  $w_{yv} \leftarrow w_{yv} + \frac{1}{n^x n_i^y} y_{ij}^{(v)} r_{ij} x_i$
Output: $\{w_{xu}\}_{u=1}^{d_x}$, $\{w_{yv}\}_{v=1}^{d_y}$.

Algorithm 2 RMLS

Input: $\{w_{xu}\}_{u=1}^{d_x}$, $\{w_{yv}\}_{v=1}^{d_y}$, d, β, γ, θ_x, θ_y.
Initialization: randomly set L_x and L_y as L_x^0 and L_y^0, $t \leftarrow 0$.
While not converged and $t \leqslant T$
  for $u = 1 : d_x$
    calculate ω_u by ${L_y^t}^\top w_{xu}$.
    calculate ${l_{xu}}^*$ using Equation (5).
  update L_x^{t+1}.
  for $v = 1 : d_y$
    calculate η_v by ${L_x^{t+1}}^\top w_{yv}$.
    calculate ${l_{yv}}^*$ using Equation (6).
  update L_y^{t+1}, $t \leftarrow t+1$
Output: L_x^t and L_y^t.

After preprocessing, we take $\{w_{xu}\}_{i=1}^{d_x}$ and $\{w_{yv}\}_{i=1}^{d_y}$ as input and iteratively optimize L_x and L_y, as described in Algorithm 2. Suppose that each w_{xu} has on average W_x nonzeros and each w_{yv} has on average W_y nonzeros, then the average time complexity of Algorithm 2 is $O(d_x W_x d + d_y W_y d)$.

In web search, it is usually the case that queries (x here) and documents (y here) are of high dimension (e.g., $> 10^6$) but extremely sparse. In other words, both c_x and c_y are small despite large d_x and d_y. Moreover, it is quite common that for each x, there are only a few y that have response with it and vice versa, rendering quite small $\tilde{n}^y$ and $\tilde{n}^x$. This situation is easy to understand in the context of web search, since for each query only a small number of documents are retrieved and viewed, and each document can only be retrieved with a few queries and get viewed. Finally, we observed that in practice, N_x and N_y are also small. For example, in web search, with the features extracted from the content of queries and documents, each word only relates to a few queries and documents. In Algorithm 2, when input vectors are sparse, $\{w_{xu}\}_{u=1}^{d_x}$ and $\{w_{yv}\}_{v=1}^{d_y}$ are also sparse, which makes W_x and W_y small. In summary, under sparse input as we often see in web search, RMLS can be implemented fairly efficiently.

4.2 Parallelization

The learning process of RMLS is still quite expensive for web scale data due to high dimensionality of x and y. Parallelization can greatly improve the speed of learning in RMLS, making it scalable enough for massive data sets.

The key in parallelizing Algorithm 1 and Algorithm 2 is that the calculation of different parameters can be executed concurrently. In Algorithm 1 there is no dependency among the calculation of different w_{xu} and w_{yv}, therefore, they can be calculated by multiple processors or multiple computers simultaneously. Similar thing can be said in the update of L_x and L_y in Algorithm 2, since different rows are updated independently. We implement a multicore version for both Algorithm 1 and Algorithm 2. Specifically, suppose that we have K processors. we randomly partition $\{1,2,\ldots,d_x\}$ and $\{1,2,\ldots,d_y\}$ into K subsets. In Algorithm 1, different processors share $\mathcal{S}$ and calculate $\{w_{xu}\}$ and $\{w_{yv}\}$ with indices in their own partition simultaneously. In Algorithm 2, when updating L_x, different processors share the same input and L_y. Rows of L_x with indices in different partitions are updated simultaneously. The same parallelization strategy is used when updating L_y.

5. Generalization Analysis

We conduct generalization analysis for matching framework (2) in this section. We first give a generalization bound for the framework, which relies on the complexity of hypothesis spaces $\mathcal{H}_x$ and $\mathcal{H}_y$. After that, we analyze the complexity of $\mathcal{H}_x$ and $\mathcal{H}_y$ for both RMLS and PLS, and give their specific bounds. The proofs of the theorems are given in Appendix B.

We formally define $D(\mathcal{S})$ as the gap between the expected objective and the empirical objective over all L_x and L_y

$$D(\mathcal{S}) \doteq \sup_{L_x,L_y} \left| \frac{1}{n^x} \sum_{i=1}^{n^x} \frac{1}{n_i^y} \sum_{j=1}^{n_i^y} r_{ij} x_i^\top L_x L_y^\top y_{ij} - \mathbb{E}_{x,y}\left(r(x,y) x^\top L_x L_y^\top y \right) \right|,$$

and bound it. With this bound, given a solution $(\hat{L}_x, \hat{L}_y)$, we can estimate its performance on unseen data (i.e., $\mathbb{E}_{x,y}\left(r(x,y)x^\top \hat{L}_x \hat{L}_y^\top y\right)$) based on its performance on observed samples. For notational simplicity, we define $f_{L_x,L_y}(x,y) \doteq r(x,y) x^\top L_x L_y^\top y$, and further assume

$$\|x\| \leqslant 1, \;\; \|y\| \leqslant 1, \;\; r(x,y) \geqslant 0, \;\; \sup_{x,y} r(x,y) \leqslant R.$$

To characterize the sparsity of inputs, we suppose that the numbers of nonzeros in x and y are bounded by m_x and m_y.

Under Assumption 1, we divide $D(\mathcal{S})$ into two parts:

1. $\sup_{L_x,L_y} \left| \frac{1}{n^x} \sum_{i=1}^{n^x} \left(\frac{1}{n_i^y} \sum_{j=1}^{n_i^y} f_{L_x,L_y}(x_i, y_{ij}) - \mathbb{E}_{y|\{x_i\}} f_{L_x,L_y}(x_i, y) \right) \right|$, denoted as $D_1(\mathcal{S})$,

2. $\sup_{L_x,L_y} \left| \frac{1}{n_x} \sum_{i=1}^{n^x} \mathbb{E}_{y|\{x_i\}} f_{L_x,L_y}(x_i, y) - \mathbb{E}_{x,y} f_{L_x,L_y}(x,y) \right|$, denoted as $D_2(\{x_i\}_{i=1}^{n^x})$.

Clearly $D(\mathcal{S}) \leqslant D_1(\mathcal{S}) + D_2(\{x_i\}_{i=1}^{n^x})$, thus we separately bound $D_1(\mathcal{S})$ and $D_2(\{x_i\}_{i=1}^{n^x})$, and finally obtain the bound for $D(\mathcal{S})$.

We first bound $D_1(\mathcal{S})$. Suppose $\sup_{x,y,L_x,L_y} \|L_x^\top x\| \, \|L_y^\top y\| \leqslant B$, and $\sup_{L_x,L_y} \|\mathrm{vec}(L_x L_y^\top)\| \leqslant C$, where B and C are constants and $\mathrm{vec}(\cdot)$ is the vectorization of a matrix. We have

Theorem 1 *Given an arbitrary small positive number* δ*, with probability at least* $1-\delta$*, the following inequality holds:*

$$D_1(\mathcal{S}) \leqslant \frac{2CR}{\sqrt{n^x n^y}} + \frac{RB\sqrt{2\log\frac{1}{\delta}}}{\sqrt{n^x n^y}},$$

where n^y *represents the harmonic mean of* $\{n_i^y\}_{i=1}^{n^x}$.

Using similar techniques we have

Theorem 2 *Given an arbitrary small positive number* δ*, with probability at least* $1-\delta$*, the following inequality holds:*

$$D_2(\{x_i\}_{i=1}^{n^x}) \leqslant \frac{2CR}{\sqrt{n^x}} + \frac{RB\sqrt{2\log\frac{1}{\delta}}}{\sqrt{n^x}}.$$

Combining Theorem 1 and Theorem 2, we are able to bound $D(\mathcal{S})$:

Theorem 3 *Given an arbitrary small positive number* δ*, with probability at least* $1-2\delta$*, the following inequality holds:*

$$D(\mathcal{S}) \leqslant (2CR + RB\sqrt{2\log\frac{1}{\delta}})(\frac{1}{\sqrt{n^x n^y}} + \frac{1}{\sqrt{n^x}}). \tag{7}$$

Equation (7) gives a general generalization bound for framework (2). Since $n^y = \frac{n^x}{\sum_{i=1}^{n^x} 1/n_i^y}$, the bound tells us that to make the gap between the empirical objective and the expected objective small enough, we not only need large n^x, but also need large n_i^y for *each* x_i, which is consistent with our intuition. The two constants B and C are dependent on the hypothesis spaces $\mathcal{H}_x$ and $\mathcal{H}_y$. Below we will analyze B and C for PLS and RMLS, and give their specific bounds based on (7).

The following two theorems give B and C for PLS and RMLS, and give their specific bounds:

Theorem 4 *Suppose that* $\mathcal{H}_x = \{L_x \mid L_x^\top L_x = \mathbb{I}_{d\times d}\}$ *and* $\mathcal{H}_y = \{L_y \mid L_y^\top L_y = \mathbb{I}_{d\times d}\}$*, then* $B = 1$ *and* $C = \sqrt{d}$*. Thus, the generalization bound for PLS is given by*

$$D(\mathcal{S}) \leqslant (2\sqrt{d}R + R\sqrt{2\log\frac{1}{\delta}})(\frac{1}{\sqrt{n^x n^y}} + \frac{1}{\sqrt{n^x}}).$$

Theorem 5 *Suppose that* $\mathcal{H}_x = \{L_x \mid ||l_{xu}|| \leqslant \lambda_x, ||l_{xu}|| \leqslant \theta_x, 1 \leqslant u \leqslant d_x\}$ *and* $\mathcal{H}_y = \{L_y \mid ||l_{yv}|| \leqslant \lambda_y, ||l_{yv}|| \leqslant \theta_y, 1 \leqslant v \leqslant d_y\}$*. If we suppose that the numbers of nonzero elements in x and y are respectively bounded by* m_x *and* m_y*, then* $B = \sqrt{m_x m_y}\min(d\lambda_x\lambda_y, \theta_x\theta_y)$ *and* $C = \sqrt{d_x d_y}\min(\lambda_x\lambda_y, \theta_x\theta_y)$*. Thus, the generalization bound for RMLS is given by*

$$D(\mathcal{S}) \leqslant (\frac{1}{\sqrt{n^x n^y}} + \frac{1}{\sqrt{n^x}}) \times (2\sqrt{d_x d_y}\min(\lambda_x\lambda_y, \theta_x\theta_y)R + \sqrt{m_x m_y}\min(d\lambda_x\lambda_y, \theta_x\theta_y)R\sqrt{2\log\frac{1}{\delta}}).$$

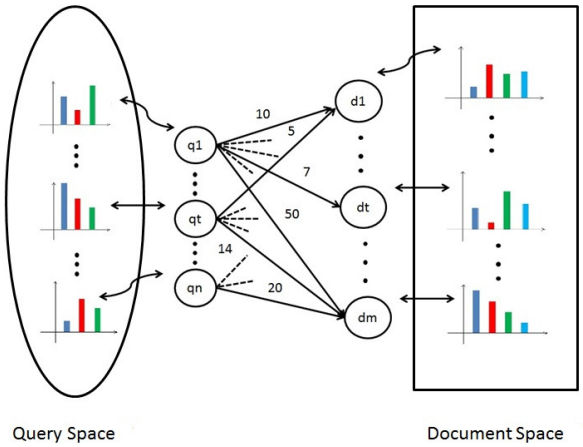


Figure 1: Click-through bipartite graph with metadata on nodes, representing queries and documents in feature spaces and their associations.

6. Experiment

We applied both RMLS and PLS to relevance ranking in web search, where matching models are used to predict relevance. Specifically, we suppose that we have a click-through bipartite with vertices representing queries and documents. The edges between query vertices and document vertices are weighted by number of user click. Besides this, we assume that there exists metadata on the vertices of the graph. The metadata represents features of queries and documents. The features may stand for the content of queries and documents and the clicks of queries and documents on the bipartite graph (Baeza-Yates and Tiberi, 2007), as seen below. Queries and documents are represented as feature vectors in the query space and the document space, respectively. Figure 1 illustrates the relationships.

We implemented the matching framework (2) and its associated RMLS and PLS using the click-through bipartite with metadata. $\mathcal{X}$ and $\mathcal{Y}$ are the query space and the document space respectively. Given a query x and a document y, we treated user click number as the response r. In this case, we actually leveraged both user clicks and features of queries and documents to perform matching. We conducted experiments on a small data set and a large data set with millions of queries and documents.

6.1 Experiment Setup

We collected 1) one week of click-through data and 2) half a year of click-through data from a commercial web search engine. After filtering out noise, there are $94,022$ queries and $111,631$ documents in the one week data set, and $6,372,254$ queries and $4,599,849$ documents in the half year data set. We extracted features from two sources, namely word and clicks. For the word feature, we represented queries and documents as tf-idf vectors (Salton and McGill, 1986) in a word space, where words are extracted from queries, URLs and the titles of documents. There are $101,904$ and $271,561$ unique words in one week data and half year data respectively. For the click feature, we followed (Baeza-Yates and Tiberi, 2007) and took the number of clicks of documents as a feature of queries, and the number of clicks of queries as a feature of documents.

Finally, we concatenated the features from the two sources to create long but extremely sparse feature vectors for both queries and documents. Note that query space and document space have different dimensions and characteristics, and should be treated as heterogeneous domains. Table 1 gives the statistics on the experiment data. Note that the notations in the table are the same as in Section 4.

6.1.1 BASELINE METHODS

We employed three different kinds of baseline methods:

- **Individual feature based model and graph based model:** We employed BM25 (Robertson et al., 1994) as a representative of feature based relevance models, and LSI (Deerwester et al., 1990) and random walk on click-through bipartite graph (Craswell and Szummer, 2007) ("RW" for short) as representatives of graph based relevance models. Particularly, we implemented two versions of LSI in this paper, one on a document-term matrix, denoted as LSI_{dt}, the other one on a query-document matrix with each element representing the click number, denoted as LSI_{qd}.

- **Combination of feature based model and graph based model:** We used models which linearly combine LSI_{qd} and random walk with BM25, denoted as LSI_{qd}+BM25 and RW+BM25, respectively.

- **Other existing models:** Besides the heuristic combination models, we also employed the bilingual topic model (BLTM) proposed by Gao et al. (2011) and supervised semantic indexing (SSI) proposed by Bai et al. (2009) as baseline methods. BLTM is the best performing model in Gao et al. (2011), and it can leverage both the user clicks and the content of queries and documents. SSI employs a hinge loss function to model pairwise preference between objects. It learns a large margin perceptron to map queries and documents into a latent space and measures their similarity in the space. To implement SSI, we additionally collected impression data from the search log. Besides click numbers, the data also contains positions of documents in ranking lists of search engine. We followed the rules proposed by Joachims (2002) to generate preference pairs.

- **Model proposed for collaborative filtering:** Besides the models in information retrieval, we also employed a state of the art model in collaborative filtering as a baseline method[3] (Chen et al., 2012). The model, named SVDFeature, can leverage the same features as RMLS and PLS.

We obtained relevance data consisting of judged query-document pairs from the search engine in a different time period from the click-through data. There are five levels of judgments, including "Perfect", "Excellent", "Good", "Fair", and "Bad". For one week data, we obtained $4,445$ judged queries and each query has on average 11.34 judged documents. For half year data, more judged data was collected. There are $57,514$ judged queries and each query has on average 13.84 judged documents. We randomly split each judged data set and used half of them for tuning model parameters and the other half for model evaluation. In summary, for both data sets, we learned models on the whole click-through data, tuned model parameters on the validation set of relevance data and evaluated model performances on the held-out test set.

3. The model won the 1st place in track 1 of KDD-Cup 2012.

	d_x	d_y	c_x	c_y	$\tilde{n}^y$	$\tilde{n}^x$	N_x	N_y
one week	$2.1 \cdot 10^5$	$2.0 \cdot 10^5$	4.0	5.9	1.74	1.46	1.7	3.4
half year	$4.9 \cdot 10^6$	$6.6 \cdot 10^6$	5.5	8.6	2.92	4.04	7.2	5.9

Table 1: Statistics on query and document features

To evaluate the performances of different methods, we employed Normalized Discounted Cumulative Gain (NDCG) (Jarvelin and Kekalainen, 2000) at positions of 1, 3, and 5 as evaluation measures.

6.2 Parameter Setting

We set the parameters for the methods in the following way. In BM25, the default setting was used. There are two parameters in random walk: the self-transition probability and the number of transition steps. Following the conclusion from Craswell and Szummer (2007), we fixed the self-transition probability as 0.9 and chose the number of transition steps from $\{1,\ldots,10\}$ on the validation set. We found that random walk reaches a "stable" state with just a few steps. In our experiments, after five steps we saw no improvement on the validation data in terms of evaluation measures. Therefore, we set five as the number of transition steps of random walk.

In LSI, SVDFeature, BLTM, SSI, PLS and RMLS, one important parameter is the dimensionality of latent space. We set the parameter in the range of $\{100,200,\ldots,1000\}$. We found that the performance of both PLS and RMLS on the validation data improves with dimensionality. On the other hand, a large dimensionality means more parameters to store in memory ($d \times (d_x + d_y)$) and more training time for each iteration. Therefore, we finally chose 1000 as the dimensionality of latent space for PLS and RMLS. For other baseline methods except SSI, a similar phenomenon was observed. For SSI, we found that its performance on the validation data is not sensitive to the dimensionality of latent space.

Besides dimensionality of latent space, parameters for regularization items (e.g., θ_x, θ_y, β, and γ in Equation (4), learning rates in SSI and SVDFeature, and number of iterations may also affect the final performance and therefore need tuning. Again, we tuned these parameters on the validation data one by one. Particularly, we found that the performance of RMLS is not sensitive to parameters for ℓ_2 norm (i.e., θ_x, θ_y). In addition, we observed that RMLS can quickly reach a good performance after a few iterations (less than 10 loops), and the early stopping also led to good generalization on the test data.

In LSI_{qd}+BM25 and RW+BM25, the combination weights are also parameters. We tuned the combination weights within $\{0.1,0.2,\ldots,0.9\}$ on the validation data.

6.3 Results on One Week Data

We conducted experiments on a workstation with 24 AMD Opteron 6172 processors and 96 GB RAM. We first compared the performance of different methods, with results summarized in Table 2. We can see that RMLS performs comparably well with PLS, and both of them significantly outperform all other baselines ($p < 0.05$ from t-test). Among the baseline methods, the performance of SSI is rather poor. We analyzed the data and found that although pairwise preference can alleviate rank bias, it also misses some important information. For example, we observed that 49.1% of 40,676 pairs that have judgments in our labeled data violate the rules proposed by Joachims (2002).

	NDCG@1	NDCG@3	NDCG@5
RMLS	**0.686**	**0.732**	0.729
PLS	0.676	**0.728**	**0.736**
SVDFeature	0.663	0.720	0.727
BLTM	0.657	0.702	0.701
SSI	0.538	0.621	0.629
RW	0.655	0.704	0.704
RW+BM25	0.671	0.718	0.716
LSI_{qd}	0.588	0.665	0.676
LSI_{qd}+BM25	0.649	0.705	0.706
LSI_{dt}	0.616	0.675	0.680
BM25	0.637	0.690	0.690

Table 2: Relevance ranking result on one week data

Documents ranked higher in these pairs have more click-through rates[4] and better or equally good judgments than documents ranked lower. Those query document associations will be well represented in either PLS or RMLS.

We then compared RMLS with PLS on efficiency. In PLS, the linear mappings are learned through SVD. We implemented an SVD solver using power method (Wegelin, 2000) with C++, and further optimized the data structure for our task. This SVD implementation can handle large data sets on which state-of-the-art SVD tools like SVDLIBC[5] fail. Since the efficiency of algorithms is influenced by implementation strategies, for example, different numbers of iterations or termination criteria, to make a fair comparison, we only report the time cost in the learning of the best performing models. RMLS significantly improves the efficiency of PLS. On a single processor, it takes RMLS $1,380$ seconds to train the model, while the training of PLS needs $945,382$ seconds. The reason is that PLS requires SVD and has a complexity of at least $O(dcd_xd_y + d^2\max(d_x,d_y))$, where c represents the density of the matrix for SVD. Even with a small c, the high dimensionality of input space (i.e., large d_x and d_y) and the quadratic growth with respect to d still make SVD quite expensive. For RMLS, W_x and W_y are quite small with a sparse input (W_x=24.82 W_y= 27.05), and hence the time complexity is nearly linear to $d \cdot \max(d_x,d_y)$. Therefore, RMLS is significantly more efficient than PLS with high dimensional but sparse inputs.

Finally, we examined the time cost of parallelized RMLS on multiple processors, as summarized by Figure 2. Clearly the running time decreases with the number of threads. With 20 threads, RMLS only takes 277 seconds to achieve a comparable performance with PLS.

6.4 Results on Half Year Data

We further tested the performance of RMLS and PLS on a half year data set with millions of queries and documents. On such large scale, SVD-based methods and random walk become infeasible (e.g., taking months to run). SSI is also infeasible because of the huge number of pairs. We therefore implemented RMLS with *full* features and PLS with only word features, and compared them with

4. Click-through rate = number of clicks / number of impressions.
5. SVDLIBC can be found at `http://tedlab.mit.edu/~dr/SVDLIBC/`.

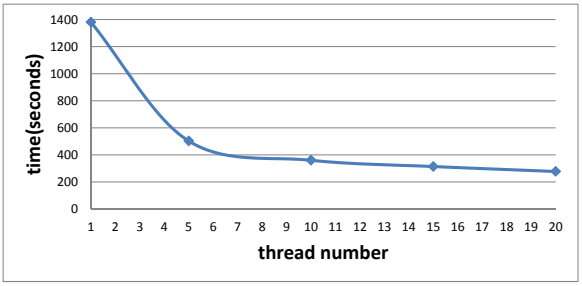


Figure 2: Time cost trend of RMLS under multiple processors.

	NDCG@1	NDCG@3	NDCG@5
RMLS	**0.742**	**0.767**	**0.776**
PLS (word)	0.638	0.666	0.677
SVDFeature	**0.742**	0.746	0.749
BLTM	0.684	0.708	0.717
BM25	0.643	0.663	0.670

Table 3: Relevance ranking result on half year data

BM25, BLTM, and SVDFeature.[6] With only word features ($d_x = d_y = 271,561$), PLS is slow but still feasible.

As shown in Table 3, RMLS outperforms all the baselines ($p < 0.01$ from t-test). We hypothesize that PLS with full features can perform comparably with RMLS, but the high computation complexity of PLS prevents us from testing it. For RMLS, it takes 20,523 seconds to achieve the result using 20 threads. For PLS, although it only uses word features, it takes 1,121,440 seconds to finish learning. In other words, parallelized RMLS can be used to tackle web search problem of real-world scale.

6.5 Discussion

In this section, we investigate the effect of matching models as features in a state of the art learning to rank algorithm and performance of matching models across queries with different numbers of click.

6.5.1 MATCHING MODELS AS FEATURES IN A LEARNING TO RANK ALGORITHM

Results in Table 2 and Table 3 demonstrate the efficacy of RMLS and PLS as individual relevance models. In modern web search, relevance models usually act as features in a learning to rank system. Therefore, a natural question is whether RMLS or PLS can enhance the performance of learn to rank algorithms as an additional feature. To answer this question, we employed RankLib[7] and trained a gradient boosted tree (MART, Friedman, 2001) with the validation data. We conducted experiments in the following three steps: first, we trained a ranker with all baseline methods as features. We denoted it as MART-Baseline. Then, we included PLS and RMLS respectively as an additional

6. SVDFeature is actually *not* based on SVD implementation.
7. RankLib can be found at `http://people.cs.umass.edu/~vdang/ranklib.html`.

	NDCG@1	NDCG@3	NDCG@5
MART-Baseline	0.661	0.737	0.779
MART-PLS	0.683	0.751	0.792
MART-RMLS	0.681	0.750	0.789
MART-All	0.689	0.757	0.797

Table 4: Performance of gradient boosted tree (MART) on one week data

	NDCG@1	NDCG@3	NDCG@5
MART-Baseline	0.708	0.760	0.791
MART-PLS	0.706	0.760	0.792
MART-RMLS	0.757	0.799	0.827
MART-All	0.756	0.798	0.826

Table 5: Performance of gradient boosted tree (MART) on half year data

feature and denoted the new rankers as MART-PLS and MART-RMLS, respectively. Finally, we trained a ranker with all models as features and denoted it as MART-All. Table 4 and Table 5 present the evaluation results.

From Table 4, we conclude that 1) RMLS and PLS significantly improve the performance of MART with baseline methods as features, which demonstrates the efficacy of the proposed framework in relevance ranking; 2) RMLS can be a good alternative to PLS in practice, because MART-PLS and MART-RMLS are comparable in ranking performance but RMLS is more efficient and scalable; 3) There is overlap between the effect of RMLS and PLS in learning to rank, because when including all models as features, the performance of ranker is only slightly improved. Results in Table 5 further demonstrate the advantage of RMLS in relevance ranking. PLS is not capable of leveraging all features of large scale data, and therefore fails to improve the performance of MART. On the other hand, RMLS successfully leverages both word features and click features, and significantly improves the ranking performance of MART.

6.5.2 EVALUATION ACROSS QUERIES WITH DIFFERENT NUMBERS OF CLICKS

In Framework (2), response r is treated as a weight for each object pair (x,y). The framework, when applied to web search, weights each query-document pair with the number of clicks between them. Usually, number of clicks has a large variance among queries, from a few to tens of thousands. An interesting question is therefore how different matching models perform across queries with different numbers of click. To answer this question, we divided queries into different bins based on the total numbers of clicks associated with them over documents. We took four levels: $totalclick \leqslant 10$, $10 < totalclick \leqslant 100$, $100 < totalclick \leqslant 1000$, and $totalclick > 1000$. We separately evaluated matching models on each level. Table 6 and Table 7 show the evaluation results, where @1, @3, and @5 mean NDCG@1, NDCG@3, and NDCG@5, respectively.

From Table 6, we can see that RMLS and PLS beat other baseline methods on queries with moderate and large number of clicks, but lose to RW and RW+BM25 when queries only have relatively few clicks (less than 100). RMLS and PLS use the absolute click number as a weight for each query-document pair. Therefore, in training, head queries may overwhelm those tail queries. We try to mitigate this effect by taking some simple transformations (e.g., logarithm) on click num-

	$totalclick \leqslant 10$			$10 < totalclick \leqslant 100$			$100 < totalclick \leqslant 1000$			$totalclick > 1000$		
	# *queries* = 230			#*queries* = 772			# *queries* = 757			#*queries* = 464		
	@1	@3	@5	@1	@3	@5	@1	@3	@5	@1	@3	@5
RMLS	0.754	**0.795**	**0.767**	0.749	0.791	0.766	**0.679**	**0.744**	**0.755**	0.557	0.612	0.655
PLS	0.704	0.776	0.764	0.706	0.769	0.748	0.650	0.727	**0.754**	**0.655**	**0.661**	**0.695**
SVDFeature	0.648	0.732	0.707	0.684	0.742	0.727	0.647	0.723	0.750	0.576	0.632	0.669
BLTM	0.758	0.787	0.755	0.750	0.795	0.766	0.636	0.700	0.726	0.485	0.557	0.603
SSI	0.633	0.730	0.700	0.607	0.696	0.673	0.523	0.616	0.657	0.403	0.491	0.540
RW	0.769	0.793	0.760	**0.773**	0.809	0.780	0.622	0.709	0.740	0.458	0.527	0.582
RW+BM25	**0.773**	0.786	0.758	0.770	**0.815**	**0.787**	0.654	0.726	0.750	0.485	0.554	0.601
LSI_{qd}	0.631	0.717	0.687	0.634	0.709	0.696	0.584	0.676	0.703	0.496	0.573	0.621
LSI_{qd}+BM25	0.696	0.745	0.719	0.726	0.777	0.759	0.646	0.716	0.736	0.500	0.576	0.619
LSI_{dt}	0.685	0.745	0.730	0.688	0.752	0.727	0.608	0.676	0.705	0.473	0.549	0.596
BM25	0.698	0.746	0.724	0.719	0.772	0.748	0.641	0.701	0.722	0.463	0.544	0.592

Table 6: Evaluation on different query bins on one week data

	$totalclick \leqslant 10$			$10 < totalclick \leqslant 100$			$100 < totalclick \leqslant 1000$			$totalclick > 1000$		
	# *queries* = 704			# *queries* = 5260			# *queries* = 8980			# *queries* = 13813		
	@1	@3	@5	@1	@3	@5	@1	@3	@5	@1	@3	@5
RMLS	**0.804**	**0.801**	**0.792**	**0.785**	**0.794**	**0.796**	**0.780**	**0.795**	**0.797**	0.698	**0.742**	**0.760**
PLS (word)	0.723	0.720	0.708	0.681	0.697	0.703	0.668	0.691	0.697	0.599	0.642	0.660
SVDFeature	0.728	0.730	0.721	0.738	0.734	0.734	0.766	0.765	0.761	**0.728**	**0.740**	0.748
BLTM	0.783	0.775	0.763	0.750	0.757	0.760	0.725	0.745	0.749	0.626	0.669	0.688
BM25	0.747	0.739	0.717	0.710	0.713	0.712	0.690	0.703	0.704	0.582	0.622	0.641

Table 7: Evaluation on different query bins on half year data

bers, but find that simple transformations not only fail to deal with tail query issues but also hurt the performance of RMLS and PLS on head queries. In contrast, BLTM and SSI perform better on tail queries than themselves on head queries. The phenomenon reminds us that introducing large margin into Framework (2) could be a potential approach to solve the problem of RMLS, although after doing so, scalability may become a more serious issue, which we leave to our future work.

SVDFeature suffers from head query effect more seriously than RMLS and PLS, which may stem from its directly fitting similarity function with absolute click numbers.

In Table 7, due to the scalability issue, results of some baseline methods are not available. In spite of this, we can still see that SVDFeature performs consistently with itself on one week data, and we can also guess that the comparisons of RMLS with RW and RW+BM25 may follow the same trends as those on one week data.

7. Conclusion

We have proposed a framework for learning to match heterogeneous data via shared latent structures, and studied its generalization ability under a hierarchical sampling assumption for web search. The framework subsumes Partial Least Squares (PLS) as a special case, and enables us to devise a more scalable algorithm called Regularized Mapping to Latent Structures (RMLS) as another special case. We applied both PLS and RMLS to web search, leveraging a click-through bipartite graph with metadata representing features of queries and documents to learn relevance models. Results on a small data set and a large data set with millions of queries and documents show the promising performance of PLS and RMLS, and particularly demonstrate the advantage of RMLS on scalability.

Appendix A. Degenerative Solution With ℓ_1 Constraints *Only*

Suppose that all the input vectors have only non-negative elements (which is natural in web search), we consider the following learning problem:

$$\begin{aligned} \arg\max_{L_x, L_y} \quad & \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} r_{ij} x_i^\top L_x L_y^\top y_{ij}, \\ s.t. \quad & |l_{xu}| \leqslant \lambda_x, \quad 1 \leqslant u \leqslant d_x, \\ & |l_{yv}| \leqslant \lambda_y, \quad 1 \leqslant v \leqslant d_y. \end{aligned} \tag{8}$$

We assert that the optimization problem (8) has a global optimum, and the global optimum can be obtained by letting $L_x = \lambda_x e_x l^\top$ and $L_y = \lambda_y e_y l^\top$, where l is a d dimensional vector satisfying $|l| = 1$ and $\|l\| = 1$, e_x is a d_x dimensional vector with all elements ones and e_y is a d_y dimensional vector with all elements ones. To demonstrate this, first we prove that the objective function (8) can be upper bounded under ℓ_1 constraints:

Proof The objective of problem (8) can be re-written as

$$\begin{aligned} \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} r_{ij} x_i^\top L_x L_y^\top y_{ij} &= \sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{r_{ij}}{n^x n_i^y}\sum_{u=1}^{d_x}\sum_{v=1}^{d_y}(l_{xu})^\top\left(x_i^{(u)} y_{ij}^{(v)} l_{yv}\right) \\ &= \sum_{u=1}^{d_x}\sum_{v=1}^{d_y}(l_{xu})^\top\left(\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{r_{ij}}{n^x n_i^y} x_i^{(u)} y_{ij}^{(v)} l_{yv}\right). \end{aligned}$$

If we define $a_{uv} = \sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{r_{ij}}{n^x n_i^y} x_i^{(u)} y_{ij}^{(v)}$, the objective of problem (8) can be re-written as

$$\sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv} l_{xu}^\top l_{yv}.$$

Since the input vectors are non-negative, $a_{uv} \geqslant 0, \forall u, v$. Thus, we have

$$\begin{aligned} \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv} l_{xu}^\top l_{yv} &\leqslant \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv}\left(\sum_{k=1}^{d}|l_{xu}^{(k)}||l_{yv}^{(k)}|\right) \\ &\leqslant \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv}\left(\max_{1\leqslant k\leqslant d}(|l_{xu}^{(k)}|)\sum_{k=1}^{d}|l_{yv}^{(k)}|\right). \end{aligned}$$

Since $|l_{xu}| \leqslant \lambda_x$ and $|l_{yv}| \leqslant \lambda_y$, we know that $\max_{1\leqslant k\leqslant d}(|l_{xu}^{(k)}|) \leqslant \lambda_x$, and thus we have

$$\sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv}\left(\max_{1\leqslant k\leqslant d}(|l_{xu}^{(k)}|)\sum_{k=1}^{d}|l_{yv}^{(k)}|\right) \leqslant \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv}\lambda_x\lambda_y.$$

■

With the existence of the upper bound, we can see that if $L_x = \lambda_x e_x l^\top$ and $L_y = \lambda_y e_y l^\top$, the value of the objective (8) is

$$\sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv} l_{xu}^\top l_{yv} = \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv}\lambda_x\lambda_y\|l\|^2 = \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} a_{uv}\lambda_x\lambda_y.$$

Thus, the optimization problem (8) reaches its global optimum when $L_x = \lambda_x e_x l^\top$ and $L_y = \lambda_y e_y l^\top$, which is an undesired degenerative solution.

Appendix B. Proofs of Theorems

We give the proofs of the theorems in Section 5.

B.1 Proof of Theorem 1

Theorem 1 *Given an arbitrary small positive number δ, with probability at least $1-\delta$, the following inequality holds:*

$$D_1(\mathcal{S}) \leqslant \frac{2CR}{\sqrt{n^x n^y}} + \frac{RB\sqrt{2\log\frac{1}{\delta}}}{\sqrt{n^x n^y}},$$

where n^y represents the harmonic mean of $\{n_i^y\}_{i=1}^{n^x}$.

To prove this theorem, we need two lemmas:

Lemma 1 *Given $\varepsilon > 0$, the following inequality holds:*

$$P\left(D_1(\mathcal{S}) - \mathbb{E}_{\{y_{ij}\}|\{x_i\}} D_1(\mathcal{S}) \geqslant \varepsilon | \{x_i\}\right) \leqslant \exp\left(-\frac{\varepsilon^2 n^x n^y}{2R^2B^2}\right).$$

Proof Given $\{x_i\}_{i=1}^{n^x}$, we re-write $D_1(\mathcal{S})$ as $D_1(\{y_{ij}\}|\{x_i\})$. Since $\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant B$ and $r \leqslant R$, $\forall\, u, v$, we have

$$|D_1(\{y_{ij}\}|\{x_i\}) - D_1\left((\{y_{ij}\} - y_{uv}) \bigcup y'_{uv} | \{x_i\}\right)| \leqslant \frac{2RB}{n^x n_u^y}.$$

Given $\{x_i\}_{i=1}^{n^x}$, $\{y_{ij}\}$ are independent. By McDiarmid inequality (Bartlett and Mendelson, 2002), we know

$$\begin{aligned}
P\left(D_1(\mathcal{S}) - \mathbb{E}_{\{y_{ij}\}|\{x_i\}} D_1(\mathcal{S}) \geqslant \varepsilon | \{x_i\}\right) &\leqslant exp\left(-\frac{2\varepsilon^2}{\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y}\frac{4R^2B^2}{\left(n^x n_i^y\right)^2}}\right) \\
&= exp\left(-\frac{\varepsilon^2}{2R^2B^2\sum_{i=1}^{n^x}\frac{1}{(n^x)^2 n_i^y}}\right) \\
&= exp\left(-\frac{\varepsilon^2 (n^x)^2}{2R^2B^2\sum_{i=1}^{n^x}\frac{1}{n_i^y}}\right) \\
&= exp\left(-\frac{\varepsilon^2 n^x n^y}{2R^2B^2}\right).
\end{aligned}$$

■

Lemma 2

$$\mathbb{E}_{\{y_{ij}\}|\{x_i\}} D_1(\mathcal{S}) \leqslant \frac{2CR}{\sqrt{n^x n^y}}.$$

Proof Define $r(x,y)x^\top L_x L_y^\top y$ as $f_{L_x,L_y}(x,y)$. We have

$$\begin{aligned}\mathbb{E}_{\{y_{ij}\}|\{x_i\}} D_1(\mathcal{S}) &= \mathbb{E}_{\{y_{ij}\}|\{x_i\}} \sup_{L_x,L_y} |\frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} f_{L_x,L_y}(x_i,y_{ij}) - \frac{1}{n^x}\sum_{i=1}^{n^x}\mathbb{E}_{y|\{x_i\}} f_{L_x,L_y}(x_i,y)| \\ &= \mathbb{E}_{\{y_{ij}\}|\{x_i\}} \sup_{L_x,L_y} |\frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} f_{L_x,L_y}(x_i,y_{ij}) - \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y}\mathbb{E}_{\{y'_{ij}\}|\{x_i\}} f_{L_x,L_y}(x_i,y'_{ij})|,\end{aligned}$$

where $\{y'_{ij}\}$ are i.i.d. random variables with $\{y_{ij}\}$.

$$\begin{aligned}&\mathbb{E}_{\{y_{ij}\}|\{x_i\}} \sup_{L_x,L_y} |\frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} f(x_i,y_{ij}) - \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y}\mathbb{E}_{\{y'_{ij}\}|\{x_i\}} f(x_i,y'_{ij})| \\ &\leqslant \mathbb{E}_{\{y_{ij}\},\{y'_{ij}\}|\{x_i\}} \sup_{L_x,L_y} \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} |f(x_i,y_{ij}) - f(x_i,y'_{ij})| \\ &= \mathbb{E}_{\{y_{ij}\},\{y'_{ij}\},\{\sigma_{ij}\}|\{x_i\}} \sup_{L_x,L_y} \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y} \sigma_{ij}\left(f(x_i,y_{ij}) - f(x_i,y'_{ij})\right),\end{aligned}$$

where given $\{x_i\}_{i=1}^{n^x}$, $\{\sigma_{ij}\}$ are i.i.d. random variables with $P(\sigma_{ij}=1) = P(\sigma_{ij}=-1) = 0.5$.

$$\mathbb{E}_{\{y_{ij},y'_{ij},\sigma_{ij}\}|\{x_i\}} \sup_{L_x,L_y} \sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y} \frac{\sigma_{ij}\left(f(x_i,y_{ij}) - f(x_i,y'_{ij})\right)}{n^x n_i^y} \leqslant 2\mathbb{E}_{\{y_{ij},\sigma_{ij}\}|\{x_i\}} \sup_{L_x,L_y} |\sum_{i=1}^{n^x}\sum_{j=1}^{n_i^y} \frac{\sigma_{ij} f(x_i,y_{ij})}{n^x n_i^y}|.$$

Note that

$$\sigma_{ij} f(x_i,y_{ij}) = \sigma_{ij} r(x_i,y_{ij}) x_i^\top L_x L_y^\top y_{ij} = \sigma_{ij}\left\langle vec(L_x L_y^\top), r(x_i,y_{ij})\mathrm{vec}(y_{ij}\otimes x_i)\right\rangle,$$

where $y_{ij}\otimes x_i$ represents the tensor of column vectors y_{ij} and x_i, and $\mathrm{vec}(\cdot)$ is the vectorization of a matrix. Thus, we have

$$\begin{aligned}&\sup_{L_x,L_y} |\frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y}\sigma_{ij} f(x_i,y_{ij})| \\ &= \sup_{L_x,L_y} |\langle \mathrm{vec}(L_x L_y^\top), \frac{1}{n^x}\sum_{i=1}^{n^x}\frac{1}{n_i^y}\sum_{j=1}^{n_i^y}\sigma_{ij} r(x_i,y_{ij})\mathrm{vec}(y_{ij}\otimes x_i)\rangle| \\ &\leqslant \sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| \sqrt{\frac{1}{(n^x)^2}\sum_{i=1}^{n^x}\sum_{j=1}^{n^x}\frac{1}{n_i^y n_j^y}\sum_{u=1}^{n_i^y}\sum_{v=1}^{n_j^y}\sigma_{iu}\sigma_{jv} r(x_i,y_{iu}) r(x_j,y_{jv})\langle x_i,x_j\rangle\langle y_{iu},y_{jv}\rangle}.\end{aligned}$$

Since we suppose that $\sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| \leqslant C$, we have

$$
\begin{aligned}
&2\mathbb{E}_{\{y_{ij}\},\{\sigma_{ij}\}|\{x_i\}} \sup_{L_x,L_y} \left| \frac{1}{n^x} \sum_{i=1}^{n^x} \frac{1}{n_i^y} \sum_{j=1}^{n_i^y} \sigma_{ij} f(x_i, y_{ij}) \right| \\
&\leqslant 2C \sqrt{\frac{1}{(n^x)^2} \sum_{i=1}^{n^x} \sum_{j=1}^{n^x} \frac{1}{n_i^y n_j^y} \sum_{u=1}^{n_i^y} \sum_{v=1}^{n_j^y} \mathbb{E}_{\{y_{ij}\},\{\sigma_{ij}\}|\{x_i\}} \left(\sigma_{iu}\sigma_{jv} r(x_i, y_{iu}) r(x_j, y_{jv}) \langle x_i, x_j \rangle \langle y_{iu}, y_{jv} \rangle \right)} \\
&= 2C \sqrt{\frac{1}{(n^x)^2} \sum_{i=1}^{n^x} \frac{1}{(n_i^y)^2} \sum_{j=1}^{n_i^y} \mathbb{E}_{\{y_{ij}\}|\{x_i\}} \left(r^2(x_i, y_{ij}) \langle x_i, x_i \rangle \langle y_{ij}, y_{ij} \rangle \right)} \\
&\leqslant 2C \sqrt{\frac{1}{(n^x)^2} \sum_{i=1}^{n^x} \frac{1}{(n_i^y)^2} n_i^y R^2} \\
&\leqslant 2CR \frac{1}{\sqrt{n^x n^y}}.
\end{aligned}
$$

We obtain the conclusion of the lemma. ■

With Lemma 1 and Lemma 2, we can prove Theorem 1:

Proof By the conclusions of Lemma 1 and Lemma 2, we have

$$
\begin{aligned}
P\left(D_1(\mathcal{S}) - \frac{2CR}{\sqrt{n^x n^y}} \geqslant \varepsilon \right) &= \mathbb{E}_{\{x_i\}} P\left(D_1(\mathcal{S}) - \frac{2CR}{\sqrt{n^x n^y}} \geqslant \varepsilon | \{x_i\} \right) \\
&\leqslant \mathbb{E}_{\{x_i\}} P\left(D_1(\mathcal{S}) - \mathbb{E}_{\{y_{ij}\}|\{x_i\}} D_1(\mathcal{S}) \geqslant \varepsilon | \{x_i\} \right) \\
&\leqslant \exp\left(-\frac{\varepsilon^2 n^x n^y}{2R^2 B^2} \right).
\end{aligned}
$$

Given a small number $\delta > 0$, by letting $\exp\left(-\frac{\varepsilon^2 n^x n^y}{2R^2B^2}\right) = \delta$, we have

$$
\begin{aligned}
-\frac{\varepsilon^2 n^x n^y}{2R^2 B^2} &= \log \delta \\
\varepsilon^2 &= \frac{2R^2 B^2 \log \frac{1}{\delta}}{n^x n^y} \\
\varepsilon &= \frac{RB\sqrt{2\log \frac{1}{\delta}}}{\sqrt{n^x n^y}}.
\end{aligned}
$$

Thus, with probability at least $1 - \delta$,

$$
D_1(\mathcal{S}) \leqslant \frac{2CR}{\sqrt{n^x n^y}} + \frac{RB\sqrt{2\log \frac{1}{\delta}}}{\sqrt{n^x n^y}}
$$

holds true. ■

B.2 Proof of Theorem 2

Theorem 2 *Given an arbitrary small positive number δ, with probability at least $1-\delta$, the following inequality holds:*

$$D_2(\{x_i\}_{i=1}^{n^x}) \leqslant \frac{2CR}{\sqrt{n^x}} + \frac{RB\sqrt{2\log\frac{1}{\delta}}}{\sqrt{n^x}}.$$

To prove Theorem 2, we also need two lemmas:

Lemma 3 *Given $\varepsilon > 0$, the following inequality holds:*

$$P\left(D_2(\{x_i\}_{i=1}^{n^x}) - \mathbb{E}_{\{x_i\}} D_2(\{x_i\}_{i=1}^{n^x}) \geqslant \varepsilon\right) \leqslant \exp\left(-\frac{\varepsilon^2 n^x}{2R^2B^2}\right).$$

Proof Similar to the proof of Lemma 1, since $\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant B$ and $r \leqslant R$, $\forall\ j$, we have

$$|D_2(\{x_i\}_{i=1}^{n^x}) - D_2\left((\{x_i\}_{i=1}^{n^x} - x_j)\bigcup x'_j\right)| \leqslant \frac{2RB}{n^x}.$$

Since $\{x_i\}_{i=1}^{n^x}$ are i.i.d. random variables, by McDiarmid inequality (Bartlett and Mendelson, 2002), we know

$$\begin{aligned} P\left(D_2(\{x_i\}_{i=1}^{n^x}) - \mathbb{E}_{\{x_i\}} D_2(\{x_i\}_{i=1}^{n^x}) \geqslant \varepsilon\right) &\leqslant exp\left(-\frac{2\varepsilon^2}{\sum_{i=1}^{n^x}\frac{4R^2B^2}{(n^x)^2}}\right) \\ &= exp\left(-\frac{\varepsilon^2 n^x}{2R^2B^2}\right). \end{aligned}$$

■

Lemma 4

$$\mathbb{E}_{\{x_i\}} D_2(\{x_i\}_{i=1}^{n^x}) \leqslant \frac{2CR}{\sqrt{n^x}}.$$

Proof Similar to the proof of Lemma 2,

$$\begin{aligned} \mathbb{E}_{\{x_i\}} D_2(\{x_i\}_{i=1}^{n^x}) &= \mathbb{E}_{\{x_i\}} \sup_{L_x,L_y} |\frac{1}{n^x}\sum_{i=1}^{n^x} \mathbb{E}_{y|\{x_i\}} r(x_i,y) x_i^\top L_x L_y^\top y - \mathbb{E}_{x,y} r(x,y) x^\top L_x L_y^\top y| \\ &= \mathbb{E}_{\{x_i\}} \sup_{L_x,L_y} |\frac{1}{n^x}\sum_{i=1}^{n^x} \mathbb{E}_{y|\{x_i\}} r(x_i,y) x_i^\top L_x L_y^\top y - \frac{1}{n^x}\sum_{i=1}^{n^x} \mathbb{E}_{\{x'_i\}}\mathbb{E}_{y|\{x'_i\}} r(x'_i,y) x_i'^\top L_x L_y^\top y| \\ &\leqslant \mathbb{E}_{\{x_i\},\{x'_i\}} \sup_{L_x,L_y} \frac{1}{n^x}\sum_{i=1}^{n^x} |\mathbb{E}_{y|\{x_i\}} r(x_i,y) x_i^\top L_x L_y^\top y - \mathbb{E}_{y|\{x'_i\}} r(x'_i,y) x_i'^\top L_x L_y^\top y| \\ &= \mathbb{E}_{\{x_i\},\{x'_i\},\{\sigma_i\}} \sup_{L_x,L_y} \frac{1}{n^x}\sum_{i=1}^{n^x} \sigma_i \left(\mathbb{E}_{y|\{x^i\}} r(x_i,y) x_i^\top L_x L_y^\top y - \mathbb{E}_{y|\{x'_i\}} r(x'_i,y) x_i'^\top L_x L_y^\top y\right), \end{aligned}$$

where $\{x_i'\}$ are i.i.d. random variables with $\{x_i\}$. $\{\sigma_i\}$ are i.i.d. random variables with $P(\sigma_i = 1) = P(\sigma_i = -1) = 0.5$.

$$\begin{aligned}
&\mathbb{E}_{\{x_i\},\{x_i'\},\{\sigma_i\}} \sup_{L_x,L_y} \frac{1}{n^x} \sum_{i=1}^{n^x} \sigma_i \left(\mathbb{E}_{y|\{x_i\}} r(x_i,y) x_i^\top L_x L_y^\top y - \mathbb{E}_{y|\{x_i'\}} r(x_i',y) x_i'^\top L_x L_y^\top y \right) \\
&\leqslant 2\mathbb{E}_{\{x_i\},\{\sigma_i\}} \sup_{L_x,L_y} |\frac{1}{n^x} \sum_{i=1}^{n^x} \sigma_i \mathbb{E}_{y|\{x_i\}} r(x_i,y) x_i^\top L_x L_y^\top y| \\
&= 2\mathbb{E}_{\{x_i\},\{\sigma_i\}} \sup_{L_x,L_y} |\mathbb{E}_{y|\{x_i\}} \langle \mathrm{vec}(L_x L_y^\top), \frac{1}{n^x} \sum_{i=1}^{n^x} \sigma_i r(x_i,y) \mathrm{vec}(y \otimes x_i) \rangle| \\
&\leqslant 2\mathbb{E}_{\{x_i\},\{\sigma_i\}} \mathbb{E}_{y|\{x_i\}} \sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| \sqrt{\frac{1}{(n^x)^2} \sum_{i=1}^{n^x} \sum_{j=1}^{n^x} \sigma_i \sigma_j r(x_i,y) r(x_j,y) \langle x_i, x_j \rangle \langle y, y \rangle}.
\end{aligned}$$

Since $\sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| \leqslant C$,

$$\begin{aligned}
&2\mathbb{E}_{\{x_i\},\{\sigma_i\}} \mathbb{E}_{y|\{x_i\}} \sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| \sqrt{\frac{1}{(n^x)^2} \sum_{i=1}^{n^x} \sum_{j=1}^{n^x} \sigma_i \sigma_j r(x_i,y) r(x_j,y) \langle x_i, x_j \rangle \langle y, y \rangle} \\
&\leqslant 2C \sqrt{\frac{1}{(n^x)^2} \sum_{i=1}^{n^x} \sum_{j=1}^{n^x} \mathbb{E}_{\{x_i\},\{\sigma_i\},y} \sigma_i \sigma_j r(x_i,y) r(x_j,y) \langle x_i, x_j \rangle \langle y, y \rangle} \\
&\leqslant 2C \sqrt{\frac{1}{(n^x)^2} \sum_{i=1}^{n^x} \mathbb{E}_{\{x_i\},y} r^2(x_i,y) \langle x_i, x_i \rangle \langle y, y \rangle} \\
&\leqslant \frac{2CR}{\sqrt{n^x}}.
\end{aligned}$$

We obtain the conclusion of the lemma. ■

With Lemma 3 and Lemma 4, we can prove Theorem 2:

Proof Combining the conclusions of Lemma 3 and Lemma 4, we have

$$\begin{aligned}
P\left(D_2(\{x_i\}_{i=1}^{n^x}) - \frac{2CR}{\sqrt{n^x}} \geqslant \varepsilon \right) &\leqslant P\left(D_2(\{x_i\}_{i=1}^{n^x}) - \mathbb{E}_{\{x_i\}} D_2(\{x_i\}_{i=1}^{n^x}) \geqslant \varepsilon \right) \\
&\leqslant \exp\left(-\frac{\varepsilon^2 n^x}{2R^2 B^2} \right).
\end{aligned}$$

Given a small number $\delta > 0$, by letting $\exp\left(-\frac{\varepsilon^2 n^x}{2R^2B^2}\right) = \delta$, we have

$$-\frac{\varepsilon^2 n^x}{2R^2B^2} = \log\delta$$
$$\varepsilon^2 = \frac{2R^2B^2\log\frac{1}{\delta}}{n^x}$$
$$\varepsilon = \frac{RB\sqrt{2\log\frac{1}{\delta}}}{\sqrt{n^x}}.$$

Thus, with probability at least $1-\delta$,

$$D_2(\{x_i\}_{i=1}^{n^x}) \leqslant \frac{2CR}{\sqrt{n^x}} + \frac{RB\sqrt{2\log\frac{1}{\delta}}}{\sqrt{n^x}}$$

holds true. ■

B.3 Proof of Theorem 4

Theorem 4 *Suppose that* $\mathcal{H}_x = \{L_x \mid L_x^\top L_x = \mathbb{I}_{d\times d}\}$ *and* $\mathcal{H}_y = \{L_y \mid L_y^\top L_y = \mathbb{I}_{d\times d}\}$, *then* $B = 1$ *and* $C = \sqrt{d}$. *Thus, the generalization bound for PLS is given by*

$$D(\mathcal{S}) \leqslant (2\sqrt{d}R + R\sqrt{2\log\frac{1}{\delta}})(\frac{1}{\sqrt{n^x n^y}} + \frac{1}{\sqrt{n^x}}).$$

Proof First, we analyze B. Remember that B is defined by $\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant B$. Suppose that $L_x = \left[l_x^1, l_x^2, \ldots, l_x^d\right]$ and $L_y = \left[l_y^1, l_y^2, \ldots, l_y^d\right]$, where $\{l_x^k\}_{k=1}^d$ and $\{l_y^k\}_{k=1}^d$ represent the columns of L_x and L_y respectively. Note that

$$||L_x^\top x||^2 = \sum_{k=1}^d \left(x^\top l_x^k\right)^2, ||L_y^\top y||^2 = \sum_{k=1}^d \left(y^\top l_y^k\right)^2.$$

Since $L_x^\top L_x = \mathbb{I}_{d\times d}$ and $L_y^\top L_y = \mathbb{I}_{d\times d}$, we have

$$||L_x^\top x||^2 \leqslant ||x||^2, ||L_y^\top y||^2 \leqslant ||y||^2.$$

Since $||x|| \leqslant 1$ and $||y|| \leqslant 1$, we have

$$\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant 1.$$

Thus, we can choose $B = 1$. Next, we analyze C. C is defined by $\sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| \leqslant C$. It is easy to see that

$$||\mathrm{vec}(L_x L_y^\top)||^2 = \mathrm{trace}(L_y L_x^\top L_x L_y^\top) = \mathrm{trace}(\mathbb{I}_{d\times d}) = d.$$

Thus,

$$\sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| = \sqrt{d}.$$

We can choose C as $\sqrt{d}$. ■

B.4 Proof of Theorem 5

Theorem 5 *Suppose that* $\mathcal{H}_x = \{L_x \mid |l_{xu}| \leqslant \lambda_x, ||l_{xu}|| \leqslant \theta_x, 1 \leqslant u \leqslant d_x\}$ *and* $\mathcal{H}_y = \{L_y \mid |l_{yv}| \leqslant \lambda_y, ||l_{yv}|| \leqslant \theta_y, 1 \leqslant v \leqslant d_y\}$. *If we suppose that the numbers of nonzero elements in x and y are respectively bounded by* m_x *and* m_y, *then* $B = \sqrt{m_x m_y}\min(d\lambda_x\lambda_y, \theta_x\theta_y)$ *and* $C = \sqrt{d_x d_y}\min(\lambda_x\lambda_y, \theta_x\theta_y)$. *Thus, the generalization bound for RMLS is given by*

$$D(\mathcal{S}) \leqslant \left(2\sqrt{d_x d_y}\min(\lambda_x\lambda_y, \theta_x\theta_y)R + \sqrt{m_x m_y}\min(d\lambda_x\lambda_y, \theta_x\theta_y)R\sqrt{2\log\frac{1}{\delta}}\right)\left(\frac{1}{\sqrt{n^x n^y}} + \frac{1}{\sqrt{n^x}}\right).$$

Proof Remember that B is defined by $\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant B$. Since

$$||L_x^\top x||^2 = ||\sum_{u=1}^{d_x} x^{(u)} l_{xu}||^2, ||L_y^\top y||^2 = ||\sum_{v=1}^{d_y} y^{(v)} l_{yv}||^2,$$

where $x = \left[x^{(1)}, x^{(2)}, \ldots, x^{(d_x)}\right]^\top$ and $y = \left[y^{(1)}, y^{(2)}, \ldots, y^{(d_y)}\right]^\top$. $\{l_{xu}\}_{u=1}^{d_x}$ and $\{l_{yv}\}_{v=1}^{d_y}$ represent the rows of L_x and L_y respectively. Suppose that $\forall u,\ l_{xu} = \left[l_{xu}^{(1)}, l_{xu}^{(2)}, \ldots, l_{xu}^{(d)}\right]^\top$ and $\forall v,\ l_{yv} = \left[l_{yv}^{(1)}, l_{yv}^{(2)}, \ldots, l_{yv}^{(d)}\right]^\top$. Since $||x|| \leqslant 1$, $||l_{xu}||^2 \leqslant \theta_x^2$, and $\#\{x^{(u)} \mid x^{(u)} \neq 0\} \leqslant m_x$, we have

$$\begin{aligned}
||L_x^\top x||^2 &= ||\sum_{u=1}^{d_x} x^{(u)} l_{xu}||^2 \\
&= \sum_{k=1}^{d}\left(\sum_{u=1}^{d_x} x^{(u)}\mathbb{1}[x^{(u)} \neq 0] l_{xu}^{(k)}\right)^2 \\
&\leqslant \sum_{k=1}^{d}\left(\sum_{u=1}^{d_x}(x^{(u)})^2\right)\left(\sum_{u=1}^{d_x}\mathbb{1}[x^{(u)} \neq 0](l_{xu}^{(k)})^2\right) \\
&= \left(\sum_{u=1}^{d_x}(x^{(u)})^2\right)\left(\sum_{k=1}^{d}\sum_{u=1}^{d_x}\mathbb{1}[x^{(u)} \neq 0](l_{xu}^{(k)})^2\right) \\
&= ||x||^2\left(\sum_{u=1}^{d_x}\mathbb{1}[x^{(u)} \neq 0]||l_{xu}||^2\right) \\
&\leqslant m_x\theta_x^2.
\end{aligned}$$

Similarly, since $||y|| \leqslant 1$, $||l_{yv}||^2 \leqslant \theta_y^2$, and $\#\{y^{(v)} \mid y^{(v)} \neq 0\} \leqslant m_y$ we have $||L_y^\top y||^2 \leqslant m_y\theta_y^2$. Thus $\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant \sqrt{m_x m_y}\theta_x\theta_y$. On the other hand, it is easy to see that

$$\begin{aligned}
||L_x^\top x||^2 &= ||\sum_{u=1}^{d_x} x^{(u)} l_{xu}||^2 \\
&= \sum_{k=1}^{d} \left(\sum_{u=1}^{d_x} x^{(u)} \mathbb{1}[x^{(u)} \neq 0] l_{xu}^{(k)} \right)^2 \\
&\leqslant \sum_{k=1}^{d} \left(\sum_{u=1}^{d_x} |x^{(u)} \mathbb{1}[x^{(u)} \neq 0] l_{xu}^{(k)}| \right)^2 \\
&\leqslant \sum_{k=1}^{d} \left(\max_{1\leqslant u\leqslant d_x} (|l_{xu}^{(k)}|) \sum_{u=1}^{d_x} \mathbb{1}[x^{(u)} \neq 0] |x^{(u)}| \right)^2 .
\end{aligned}$$

Since $|l_{xu}| = \sum_{k=1}^{d} |l_{xu}^{(k)}| \leqslant \lambda_x$, $\forall\ 1 \leqslant u \leqslant d_x$, thus $\max_{1\leqslant k\leqslant d} \max_{1\leqslant u\leqslant d_x} (|l_{xu}^{(k)}|) \leqslant \lambda_x$. Note that $||x|| \leqslant 1$ and $\#\{x^{(u)} \mid x^{(u)} \neq 0\} \leqslant m_x$, then we have

$$\begin{aligned}
\sum_{k=1}^{d} \left(\max_{1\leqslant u\leqslant d_x} (|l_{xu}^{(k)}|) \sum_{u=1}^{d_x} \mathbb{1}[x^{(u)} \neq 0] |x^{(u)}| \right)^2 &\leqslant \lambda_x^2 \sum_{k=1}^{d} \left(\sum_{u=1}^{d_x} \mathbb{1}[x^{(u)} \neq 0] |x^{(u)}| \right)^2 \\
&\leqslant \lambda_x^2 \sum_{k=1}^{d} \left(\sum_{u=1}^{d_x} (\mathbb{1}[x^{(u)} \neq 0])^2 \right) \left(\sum_{u=1}^{d_x} (x^{(u)})^2 \right) \\
&\leqslant d\lambda_x^2 m_x .
\end{aligned}$$

Similarly, since $||y|| \leqslant 1$, $|l_{yv}| \leqslant \lambda_y$, $\forall\ 1 \leqslant v \leqslant d_y$, and $\#\{y^{(v)} \mid y^{(v)} \neq 0\} \leqslant m_y$, we have $||L_y^\top y||^2 \leqslant d\lambda_y^2 m_y$. Thus, $\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant \sqrt{m_x m_y} d\lambda_x\lambda_y$.

Therefore, we know $\sup_{x,y,L_x,L_y} ||L_x^\top x||||L_y^\top y|| \leqslant \sqrt{m_x m_y}\min(d\lambda_x\lambda_y, \theta_x\theta_y)$, and we can choose B as $\sqrt{m_x m_y}\min(d\lambda_x\lambda_y, \theta_x\theta_y)$.

Next, we analyze C. C is defined by $\sup_{L_x,L_y} ||\mathrm{vec}(L_x L_y^\top)|| \leqslant C$. Since $||l_{xu}|| \leqslant \theta_x$ and $||l_{yv}|| \leqslant \theta_y$, $\forall\ 1 \leqslant u \leqslant d_x$ and $1 \leqslant v \leqslant d_y$, we have

$$\begin{aligned}
||\mathrm{vec}(L_x L_y^\top)||^2 &= \mathrm{trace}(L_y L_x^\top L_x L_y^\top) \\
&= \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} \left(\sum_{k=1}^{d} l_{xu}^{(k)} l_{yv}^{(k)} \right)^2 \\
&\leqslant \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} \left(\sum_{k=1}^{d} \left(l_{xu}^{(k)} \right)^2 \right) \left(\sum_{k=1}^{d} \left(l_{yv}^{(k)} \right)^2 \right) \\
&= \sum_{u=1}^{d_x}\sum_{v=1}^{d_y} ||l_{xu}||^2 ||l_{yv}||^2 \\
&\leqslant d_x d_y \theta_x^2 \theta_y^2 .
\end{aligned}$$

On the other hand, since $|l_{xu}| \leqslant \lambda_x$ and $|l_{yv}| \leqslant \lambda_y$, $\forall\ 1 \leqslant u \leqslant d_x$ and $1 \leqslant v \leqslant d_y$, we have

$$\begin{aligned}
||\mathrm{vec}(L_x L_y^\top)||^2 &= \mathrm{trace}(L_y L_x^\top L_x L_y^\top) \\
&= \sum_{u=1}^{d_x}\sum_{v=1}^{d_y}\left(\sum_{k=1}^{d} l_{xu}^{(k)} l_{yv}^{(k)}\right)^2 \\
&\leqslant \sum_{u=1}^{d_x}\sum_{v=1}^{d_y}\left(\sum_{k=1}^{d} |l_{xu}^{(k)} l_{yv}^{(k)}|\right)^2 \\
&\leqslant \sum_{u=1}^{d_x}\sum_{v=1}^{d_y}\left(\max_{1\leqslant k\leqslant d}(|l_{xu}^{(k)}|)\sum_{k=1}^{d}|l_{yv}^{(k)}|\right)^2 \\
&\leqslant \sum_{u=1}^{d_x}\sum_{v=1}^{d_y}\lambda_x^2\left(\sum_{k=1}^{d}|l_{yv}^{(k)}|\right)^2 \\
&\leqslant d_x d_y \lambda_x^2 \lambda_y^2.
\end{aligned}$$

Therefore, we have $\sup_{L_x, L_y} ||\mathrm{vec}(L_x L_y^\top)|| \leqslant \sqrt{d_x d_y}\min(\lambda_x\lambda_y, \theta_x\theta_y)$, and we can choose C as $\sqrt{d_x d_y}\min(\lambda_x\lambda_y, \theta_x\theta_y)$. ■

References

J. Abernethy, F. Bach, T. Evgeniou, and J.P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *JMLR '09*, 10:803–826, 2009.

S. Agarwal and P. Niyogi. Stability and generalization of bipartite ranking algorithms. In *COLT'05*, pages 32–47, 2005.

F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *NIPS'08*, pages 105–112, 2008.

F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *ICML'04*, 2004.

R. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs. In *SIGKDD'07*, pages 76–85, 2007.

B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, O. Chapelle, and K. Weinberger. Supervised semantic indexing. In *CIKM'09*, pages 187–196, 2009.

P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *JMLR'02*, 3:463–482, 2002.

C. Burges, R. Ragno, and Q. Le. Learning to rank with nonsmooth cost functions. In *NIPS'06*, pages 395–402. 2006.

Y. Cao, J. Xu, T.Y. Liu, H. Li, Y. Huang, and H.W. Hon. Adapting ranking svm to document retrieval. In *SIGIR '06*, pages 186–193, 2006.

Z. Cao, T. Qin, T.Y. Liu, M.F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *ICML '07*, pages 129–136, 2007.

T.Q. Chen, W.N. Zhang, Q.X. Lu, K.L. Chen, Z. Zhao, and Y. Yu. Svdfeature: A toolkit for feature-based collaborative filtering. *JMLR'12*, 13, 2012.

W. Chen, T.Y. Liu, and Z. Ma. Two-layer generalization analysis for ranking using rademacher average. *NIPS'10*, 23:370–378, 2010.

C. Cortes. Invited talk: Can learning kernels help performance? In *ICML'09*, page 161, 2009.

K. Crammer and Y. Singer. Pranking with ranking. In *NIPS'01*, pages 641–647, 2001.

N. Craswell and M. Szummer. Random walks on the click graph. In *SIGIR'07*, pages 239–246, 2007.

N. Cristianini, J. Shawe-taylor, A. Elisseeff, and J. Kandola. On kernel-target alignment. In *NIPS'01*, 2001.

J.V. Davis, B. Kulis, P. Jain, S. Sra, and I.S. Dhillon. Information-theoretic metric learning. In *ICML'07*, page 216, 2007.

S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *JASIS'90*, 41:391–407, 1990.

J.H. Friedman. Greedy function approximation: a gradient boosting machine. *Ann. Statist*, 29(5): 1189–1232, 2001.

J. Gao, K. Toutanova, and W. Yih. Clickthrough-based latent semantic models for web search. In *SIGIR'11*, pages 675–684, 2011.

D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *IEEE Transactions on PAMI*, 30(8):1371–1384, 2008.

D.R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004.

R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. *NIPS'99*, pages 115–132, 1999.

T. Hertz, A. Bar-Hillel, and D. Weinshall. Boosting margin based distance functions for clustering. In *ICML'04*, page 50, 2004.

T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22:89–115, 2004.

S. CH. Hoi, W. Liu, M.R. Lyu, and W.Y. Ma. Learning distance metrics with contextual constraints for image retrieval. In *CVPR'06*, volume 2, pages 2072–2078, 2006.

S. JacobGoldberger and R. GeoffHinton. Neighbourhood components analysis. *NIPS'04*, 2004.

K. Jarvelin and J. Kekalainen. Ir evaluation methods for retrieving highly relevant documents. In *SIGIR'00*, pages 41–48, 2000.

T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02*, pages 133–142, 2002.

I.T. Jolliffe. *Principal Component Analysis*. Springer verlag, 2002.

G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. In *ICML'02*, pages 323–330, 2002.

H. Li. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 4(1):1–113, 2011.

T.Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

H. Ma, H.X. Yang, I. King, and M. R. Lyu. Learning latent semantic relations from clickthrough data for query suggestion. In *CIKM'08*, pages 709–718, 2008.

C. Micchelli and M. Pontil. Learning the kernel function via regularization. *JMLR'05*, 6:1099–1125, 2005.

C.S. Ong, A.J. Smola, and R.C. Williamson. Learning the kernel with hyperkernels. *JMLR '05*, 6: 1043–1071, 2005.

J.M. Ponte and W.B. Croft. A language modeling approach to information retrieval. In *SIGIR'98*, pages 275–281, 1998.

S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *TREC*, 1994.

R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. *Subspace, Latent Structure and Feature Selection*, pages 34–51, 2006.

C. Rudin, C. Cortes, M. Mohri, and R. E. Schapire. Margin-based ranking meets boosting in the middle. In *COLT'05*, pages 63–78, 2005.

G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

P.J. Schreier. A unifying discussion of correlation analysis for complex random vectors. *Signal Processing, IEEE Transactions on*, 56(4):1327–1336, 2008.

M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. *NIPS'03*, 2003.

N. Srebro, J.D.M. Rennie, and T. Jaakkola. Maximum-margin matrix factorization. *NIPS'05*, pages 1329–1336, 2005.

M. Sugiyama. Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis. *JMLR'07*, 8:1027–1061, 2007.

M. Varma and B. R. Babu. More generality in efficient multiple kernel learning. In *ICML'09*, page 134, 2009.

J.A. Wegelin. A survey of partial least squares (pls) methods, with emphasis on the two-block case. *Technical Report, No.371, Seattle: Department of Statistics, Univ. of Wash.*, 2000.

K.Q. Weinberger and L.K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR'09*, 10:207–244, 2009.

W. Wu, J. Xu, H. Li, and O. Satoshi. Learning a robust relevance model for search using kernel methods. *JMLR'11*, 12:1429–1458, 2011.

E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. *NIPS'03*, pages 521–528, 2003.

J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *SIGIR' 07*, pages 391–398, 2007.

J. Xu, H. Li, and Z.L. Zhong. Relevance ranking using kernels. In *AIRS '10*, 2010.

L. Yang, R. Jin, R. Sukthankar, and Y. Liu. An efficient algorithm for local distance metric learning. In *AAAI'06*, page 543, 2006.

Y. Ying, K. Huang, and C. Campbell. Sparse Metric Learning via Smooth Optimization. *NIPS'09*, pages 521–528, 2009.

C.X. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.

Journal of Machine Learning Research 14 (2013) 2549-2582 Submitted 1/13; Revised 5/13; Published 9/13

One-shot Learning Gesture Recognition from RGB-D Data Using Bag of Features

Jun Wan 09112088@BJTU.EDU.CN
Qiuqi Ruan QQRUAN@CENTER.NJTU.EDU.CN
Wei Li 08112050@BJTU.EDU.CN
Institute of Information Science
Beijing Jiaotong University
Beijing, 100044, China

Shuang Deng DAISY_SHUANG@HOTMAIL.COM
China machinery TDI international engineering co., Itd
Beijing, 100083, China

Editors: Isabelle Guyon and Vassilis Athitsos

Abstract

For one-shot learning gesture recognition, two important challenges are: how to extract distinctive features and how to learn a discriminative model from only one training sample per gesture class. For feature extraction, a new spatio-temporal feature representation called 3D enhanced motion scale-invariant feature transform (3D EMoSIFT) is proposed, which fuses RGB-D data. Compared with other features, the new feature set is invariant to scale and rotation, and has more compact and richer visual representations. For learning a discriminative model, all features extracted from training samples are clustered with the k-means algorithm to learn a visual codebook. Then, unlike the traditional bag of feature (BoF) models using vector quantization (VQ) to map each feature into a certain visual codeword, a sparse coding method named simulation orthogonal matching pursuit (SOMP) is applied and thus each feature can be represented by some linear combination of a small number of codewords. Compared with VQ, SOMP leads to a much lower reconstruction error and achieves better performance. The proposed approach has been evaluated on ChaLearn gesture database and the result has been ranked amongst the top best performing techniques on ChaLearn gesture challenge (round 2).

Keywords: gesture recognition, bag of features (BoF) model, one-shot learning, 3D enhanced motion scale invariant feature transform (3D EMoSIFT), Simulation Orthogonal Matching Pursuit (SOMP)

1. Introduction

Human gestures frequently provide a natural and intuitive communication modality in daily life, and the techniques of gesture recognition can be widely applied in many areas, such as human computer interaction (HCI) (Pavlovic et al., 1997; Zhu et al., 2002), robot control (Malima et al., 2006; Shan et al., 2007), sign language recognition (Gao et al., 2004; T. Starner and Pentland, 1998) and augmented reality (Reifinger et al., 2007). To model gesture signals and achieve acceptable recognition performance, the most common approaches are to use Hidden Markov Models (HMMs) or its variants (Kim et al., 2007) which are a powerful model that includes hidden state structure. Yamato

et al. (1992) used image preprocessing operations (background subtraction, image blurring) to extract low-level features and used HMM to recognize tennis motions. Brand et al. (1997) suggested a coupled HMM that combined two HMMs with causal possibly asymmetric links to recognize gestures. Vogler (2003) presented a parallel HMM algorithm to model gesture components and can recognize continuous gestures in sentences. Then a more general probabilistic model named dynamic Bayesian network (DBN) is proposed. DBN includes HMMs and Kalman filters as special cases (Suk et al., 2010). Youtian et al. (2006) defined five classes of gestures for HCI and developed a DBN-based model which used local features (contour, moment, height) and global features (velocity, orientation, distance) as observations. Suk et al. (2010) proposed a DBN-based system to control media player or slide presentation. They used local features (location, velocity) by skin extraction and motion tracking to design the DBN inference.

However, both HMM and DBN models assume that observations given the motion class labels are conditional independent. This restriction makes it difficult or impossible to accommodate long-range dependencies among observations or multiple overlapping features of the observations (Sminchisescu et al., 2005). Therefore, Sminchisescu et al. (2005) proposed conditional random fields (CRF) which can avoid the independence assumption between observations and allow non-local dependencies between state and observations. Wang et al. (2006) then incorporated hidden state variables into the CRF model, namely, hidden conditional random field (HCRF). They used HCRF to recognize gesture recognition and proved that HCRF can get better performance. Later, the latent-dynamic conditional field (LDCRF) model (Morency et al., 2007) was proposed, which combines the strengths of CRFs and HCRFs by capturing both extrinsic dynamics and intrinsic sub-structure. The detailed comparisons are evaluated by Morency et al. (2007).

Another important approach is dynamic time warping (DTW) widely used in gesture recognition. Early DTW-based methods were applied to isolated gesture recognition (Corradini, 2001; Lichtenauer et al., 2008). Then Ruiduo et al. (2007) proposed an enhanced Level-Building DTW method. This method can handle the movement epenthesis problem and simultaneously segment and match signs to continuous sign language sentences. Besides these methods, other approaches are also widely used for gesture recognition, such as linguistic sub-units (Cooper et al., 2012) and topology-preserving self-organizing networks (Flórez et al., 2002). Although the mentioned methods have delivered promising results, most of them assume that the local features (shape, velocity, orientation, position or trajectory) are detected well. However, the prior successes of hand detection and tracking are major challenging problems in complex surroundings. Moreover, as shown in Table 1, most of the mentioned methods need dozens or hundreds of training samples to achieve high recognition rates. For example, in Yamato et al. (1992), the authors used at least 50 samples for each class to train HMM and got the average recognition rate 96%. Besides, Yamato et al. (1992) suggested that the recognition rate will be unstable if the number of samples is small. When there is only one training sample per class, those methods are difficult to satisfy the requirement of high performance application systems.

In recent years, BoF-based methods derived from object categories (Fei-Fei and Perona, 2005) and action recognition (Wang et al., 2009) have become an important branch for gesture recognition. Dardas and Georganas (2011) proposed a method for real-time hand gesture recognition based on standard BoF model, but they first needed to detect and track hands and that would be difficult in a clutter background. For example, when the hand and face are overlapped or the background is similar to skin color, hand detection may fail. Shen et al. (2012) extracted maximum stable extremal regions (MSER) features (Forssen and Lowe, 2007) from the motion divergence fields which

paper/method	Kim et al., 2007/HMM	Yamato et al., 1992/HMM	Youtian et al., 1992/DBN	Suk et al., 2010/DBN	Sminchisescu et al., 2005/CRF
training samples per class	150	$\geq$50	15	42	NA
paper/method	Wang et al. 2006/HCRF	Morency et al. 2007/LDCRF	Corradini 2001/DTW	Lichtenauer et al. 2008/DTW	Ruiduo et al. 2007/DTW
training samples per class	$\geq$45	$\geq$269	45	$\geq$60	NA

Table 1: This tables shows the training samples pre class needed in some traditional methods. "NA" means the training samples are not clearly mentioned.

were calculated by optical flow (Lowe, 2004), and learned a codebook using hierarchical k-means algorithm, then matched the test gesture sequence with the database using a term frequency-inverse document frequency (tf-idf) weighting scheme. These methods need dozens or hundreds of training samples. However, in this paper, we explore one-shot learning gesture recognition (Malgireddy et al., 2012), that is, using one training sample per each class. Some important challenging issues for one-shot learning gesture recognition are the following:

1. **How to extract distinctive features?** Different people have different speeds, trajectories and spatial positions to perform the same gesture. Even when a single person performs the gestures, the trajectories are not identical. Therefore, the extracted spatio-temporal features should be invariant to image-plane rotation, scale and spatial position. Simple descriptors, such as motion trajectories (Yang et al., 2002) and spatio-temporal gradients (Freeman and Roth, 1995), may not meet the invariant conditions. Therefore, we propose a new spatio-temporal feature which is scale, image-plane rotation and space invariant and can capture more compact and richer visual representations. The new feature will be introduced in Section 3.1.

2. **How to select a suitable model?** Here, we select BoF-based model to recognize gestures because it reveals promising results for one-shot learning (Hernández-Vela et al., 2012) and has a number of attractive properties. First, in our BoF representation, we do not need the prior success of hand detection and tracking. Second, BoF is a modular system with three parts, namely, i) spatio-temporal feature extraction, ii) codebook learning and descriptor coding, iii) classifier, each of which can be easily replaced with different methods. For instance, we can apply various methods, such as Cuboid (Dollár et al., 2005) or Harris3D (Laptev, 2005) for the local spatio-temporal feature extraction while leaving the rest of the system unchanged.

In this paper, we focus on solving these two challenging issues and propose a new approach to achieve good performance for one-shot learning gesture recognition. Our experimental results reveal that our method is competitive to the state-of-the-art methods. The key contributions of the proposed method are summarized as follows:

- A new framework derived from the BoF model is proposed.
- A new spatio-temporal feature (3D EMoSIFT) is proposed.
- The new feature is invariant to scale and rotation.
- The new feature is not sensitive to slight motion.
- Using SOMP instead of VQ in the coding stage.

- Obtained high ranking results on ChaLearn gesture challenge.

The rest of paper is organized as follows: Section 2 reviews the background including BoF model and some local spatio-temporal features. In Section 3, we describe the proposed approach in detail. Section 4 presents the experimental results. In Section 5, we conclude the paper and discuss future work.

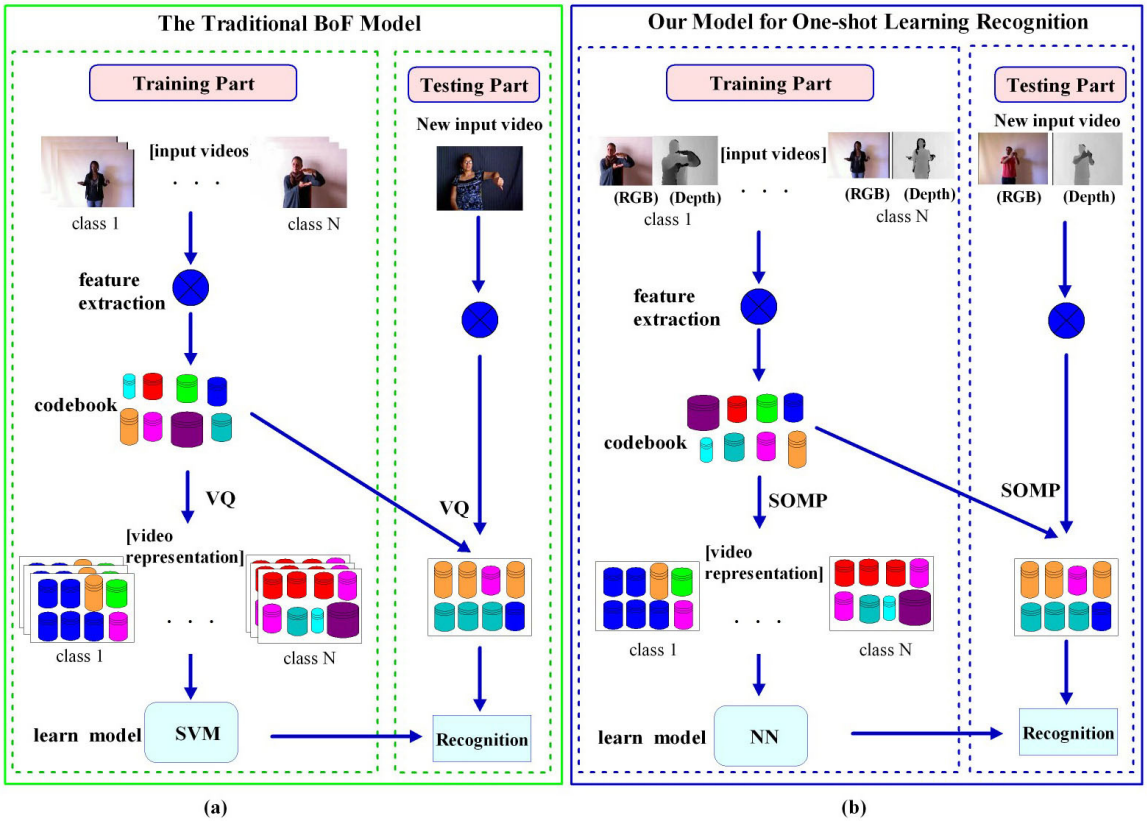


Figure 1: (a) An overview of the traditional BoF model (the left green rectangle); (b) An overview of our model (the right blue rectangle).

2. Background

In this section, we first introduce the traditional BoF framework for recognition and then review the local spatio-temporal features which are widely used in BoF model.

2.1 Traditional Bag of Feature (BoF) Model

Figure 1(a) illustrates the traditional BoF approach for gesture (or action) recognition. In the training part, after extracting local features from training videos, the visual codebook is learned with the k-means algorithm. Then each feature is mapped to a certain visual codeword through the clustering process and the video can be represented by the histogram of visual codewords. The histograms

representing training videos are treated as input vectors for a support vector machine (SVM) (Chang and Lin, 2011) to build a classifier. In the testing stage, the features are extracted from a new input video, and then those features are mapped into a histogram vector by the descriptor coding method (e.g., VQ) using the pre-trained codebook. Then, the histogram vector is finally fed into an SVM classifier to get the recognition result.

However, as shown in Figure 1(b), we list at least three differences between our model and the traditional BoF model. First, there is only one training sample per gesture class, while dozens or hundreds of training samples per class are provided in the traditional BoF model. Second, we use SOMP to replace VQ in the coding stage. That is because SOMP can get better performance. Third, in the recognition stage, we just use the simple nearest neighbor (NN) classifier instead of SVM to recognize gestures.

2.2 Spatio-Temporal Features

We describe some spatio-temporal features which represent the state-of-the-art techniques on object recognition tasks. Those features are commonly used to detect salient and stable local batches from videos.

The Cuboid detector depends on a set of linear filters for computing the response function of a video clip. The response function has the form of a 2D Gaussian smoothing function (applied in the spatial domain) and a quadrature pair of 1D Gabor filters (applied in the temporal direction). Then the keypoints are detected at the local maxima of the response function. The video batches extracted at each of the keypoints are converted to a descriptor. There are a number of ways to compute descriptors from video batches as discussed by Dollár et al. (2005). Among those, gradient-based descriptors such as histograms of oriented gradients (HOG) and concatenated gradient vectors are the most reliable ones. For more details about the Cuboid feature, please see Dollár et al. (2005).

The Harris3D detector (Laptev, 2005) is an extension of the Harris corner detector (Harris and Stephens, 1988). The author computes a spatio-temporal second-moment matrix at each video point using independent spatial and temporal scale values, a separable Gaussian smoothing function, and space-time gradients. The final locations of space-time interest points are given by the local positive spatio-temporal maxima. Then, at each keypoint, two types of descriptors are calculated, which are HOG and histograms of optical flow (HOF) descriptors.

The MoSIFT (Chen and Hauptmann, 2009) is derived from scale invariant feature transform (SIFT) (Lowe, 2004) and optical flow (Lucas et al., 1981). First, a pair of Gaussian pyramids are built from two successive frames, respectively. Then, optical flow pyramids are calculated by each layer of the pair of Gaussian pyramids. Next, a local extreme detected from difference of Gaussian pyramids (DoG) can only become an interest point if it has sufficient motion in the optical flow pyramid. Finally, as the process of the SIFT descriptor calculation, the MoSIFT descriptors are respectively computed from Gaussian pyramid and optical flow pyramid so that each MoSIFT descriptor now has 256 dimensions.

Ming et al. (2012) propose a new feature called 3D MoSIFT that is derived from MoSIFT. Compared with MoSIFT, 3D MoSIFT fuses the RGB data and depth information into the feature descriptors. First, Ming et al. (2012) adopt the same strategy using the RGB data to detect interest points. Then, for each interest point, 3D gradient space and 3D motion space are constructed by using RGB data and depth information. In 3D gradient (motion) space, they map 3D space into

three 2D planes: xy plane, yz plane and xz plane. Next, for each plane, they used SIFT algorithm to calculate the descriptors. Therefore, each 3D MoSIFT descriptor has 768 dimensions.

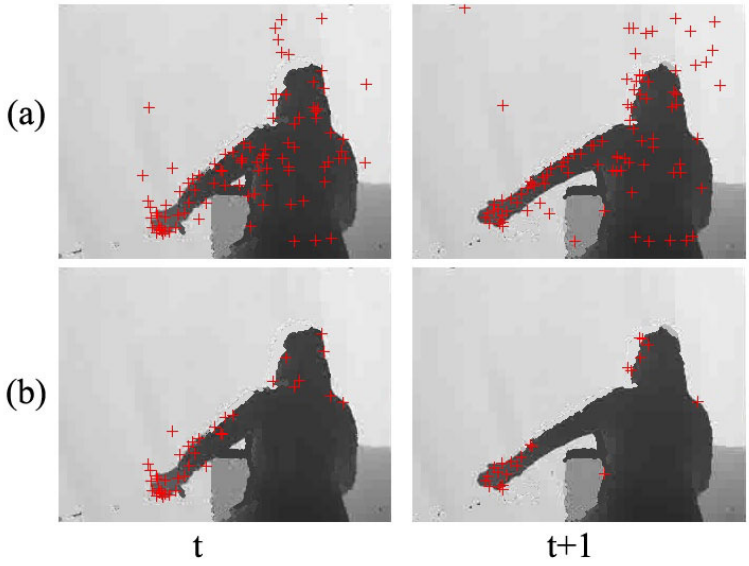


Figure 2: Results of interest point detection (marked with the red cross) in two consecutive frames. (a) 3D MoSIFT; (b) 3D EMoSIFT. We can see that some redundant points are detected in some slight motion regions (i.e., background regions) which shows 3D MoSIFT is sensitive to slight movement. However, 3D EMoSIFT can detect interest points from the regions with large motion (i.e., hand and arm regions), which shows 3D EMoSIFT is not sensitive to slight motion.

3. The Proposed Approach for One-Shot Learning Gesture Recognition

We propose a new spatio-temporal feature called 3D EMoSIFT. The new feature is invariant to scale and image-plane rotation. Then we use kmeans algorithm to learn codebook and apply SOMP algorithm to achieve descriptor coding. Besides, we adopt a methodology based on DTW and motion energy for temporal segmentation. Below, we describe each stage in detail.

3.1 Spatio-Temporal Feature Extraction: 3D EMoSIFT

The first stage is to extract rich spatio-temporal representations from the video clips. To obtain such representations, there are many ways to select (Dollár et al., 2005; Laptev, 2005; Chen and Hauptmann, 2009). However, those approaches only rely on RGB data and do not consider the depth information, which may lead to acquire insufficient information. Although 3D MoSIFT can fuse the RGB-D data to calculate descriptors, it still cannot accurately detect interest points. For instance, as shown in Figure 2(a), 3D MoSIFT capture some redundant interest points when some slight motion happens (e.g., slight motion in the background), showing that 3D MoSIFT is sensitive to slight movement. Besides, 3D MoSIFT (Ming et al., 2012) is a little sketchy. To solve the

mentioned problems, we propose a new spatio-temporal feature and give examples to explain how to extract the new feature step by step.

3.1.1 FEATURE POINTS DETECTION FROM RGB-D DATA

Although the 3D MoSIFT feature has achieved good results in human activity recognition, it still cannot eliminate some influences from the slight motion as shown in Figure 2(a). Therefore, we fuse depth information to detect robust interest points. We know that SIFT algorithm (Lowe, 2004) uses the Gaussian function as the scale-space kernel to produce a scale space of an input image. The whole scale space is divided into a sequence of octaves and each octave consists of a sequence of intervals, where each interval is a scaled image.

Building Gaussian Pyramid. Given a gesture sample including two videos (one for RGB video and the other for depth video),[1] a Gaussian pyramid for every grayscale frame (converted from RGB frame) and a depth Gaussian pyramid for every depth frame can be built via Equation (1).

$$\begin{aligned} L^I_{i,j}(x,y) &= G(x,y,k^j\sigma) * L^I_{i,0}(x,y),\ 0 \le i < n, 0 \le j < s+3, \\ L^D_{i,j}(x,y) &= G(x,y,k^j\sigma) * L^D_{i,0}(x,y),\ 0 \le i < n, 0 \le j < s+3, \end{aligned} \tag{1}$$

where (x,y) is the coordinate in an image; n is the number of octaves and s is the number of intervals; $L^I_{i,j}$ and $L^D_{i,j}$ denote the blurred image of the $(j+1)^{th}$ image in the $(i+1)^{th}$ octave; $L^I_{i,0}$ (or $L^D_{i,0}$) denotes the first grayscale (or depth) image in the $(i+1)^{th}$ octave; For $i=0$, $L^I_{0,0}$ (or $L^D_{0,0}$) is calculated from the original grayscale (depth) frame via bilinear interpolation and the size of $L^I_{0,0}$ is twice the size of the original frame; For $i>1$, $L^I_{i,0}$ (or $L^D_{i,0}$) is down-sampled from $L^I_{i-1,s}$ (or $L^D_{i-1,s}$) by taking every second pixel in each row and column. In Figure 3(a), the blue arrow shows that the first image $L^I_{1,0}$ in the second octave is down-sampled from the third image $L^I_{0,2}$ in the first octave. $*$ is the convolution operation; $G(x,y,k^j\sigma) = \frac{1}{2\pi(k^j\sigma)^2} e^{-(x^2+y^2)/(2(k^j\sigma)^2)}$ is a Gaussian function with variable-scale value; σ is the initial smoothing parameter in Gaussian function and $k = 2^{1/s}$ (Lowe, 2004). Then, the difference of Gaussian (DoG) images, Df, are calculated from the difference of two nearby scales in Equation (2).

$$Df_{i,j} = L^I_{i,j+1} - L^I_{i,j},\ 0 \le i < n, 0 \le j < s+2. \tag{2}$$

We give an example to intuitively understand the Gaussian pyramid and DoG pyramid. Figure 3 shows two Gaussian pyramids (L^{I_t}, $L^{I_{t+1}}$) built from two consecutive grayscale frames and two depth Gaussian pyramids (L^{D_t}, $L^{D_{t+1}}$) built from the corresponding depth frames. In this example, the number of octaves is $n=4$ and the number of intervals is $s=2$; Therefore, for each frame, we can build five images for each octave. And we can see that larger $k^j\sigma$ results in a more blurred image (see the enlarged portion of the red rectangle in Figure 3). Then, we use the Gaussian pyramid shown in Figure 3(a) to build the DoG pyramid via Equation (2), which is shown in Figure 4.

Building Optical Flow Pyramid. First, we briefly review the Lucas-Kanade method (Lucas et al., 1981) which is widely used in computer vision. The method assumes that the displacement of two consecutive frames is small and approximately constant within a neighborhood of the point ρ. The two consecutive frames are denoted by $F1$ and $F2$ at time t and $t+1$, respectively. Then

1. The depth values are normalized to [0 255] in depth videos.

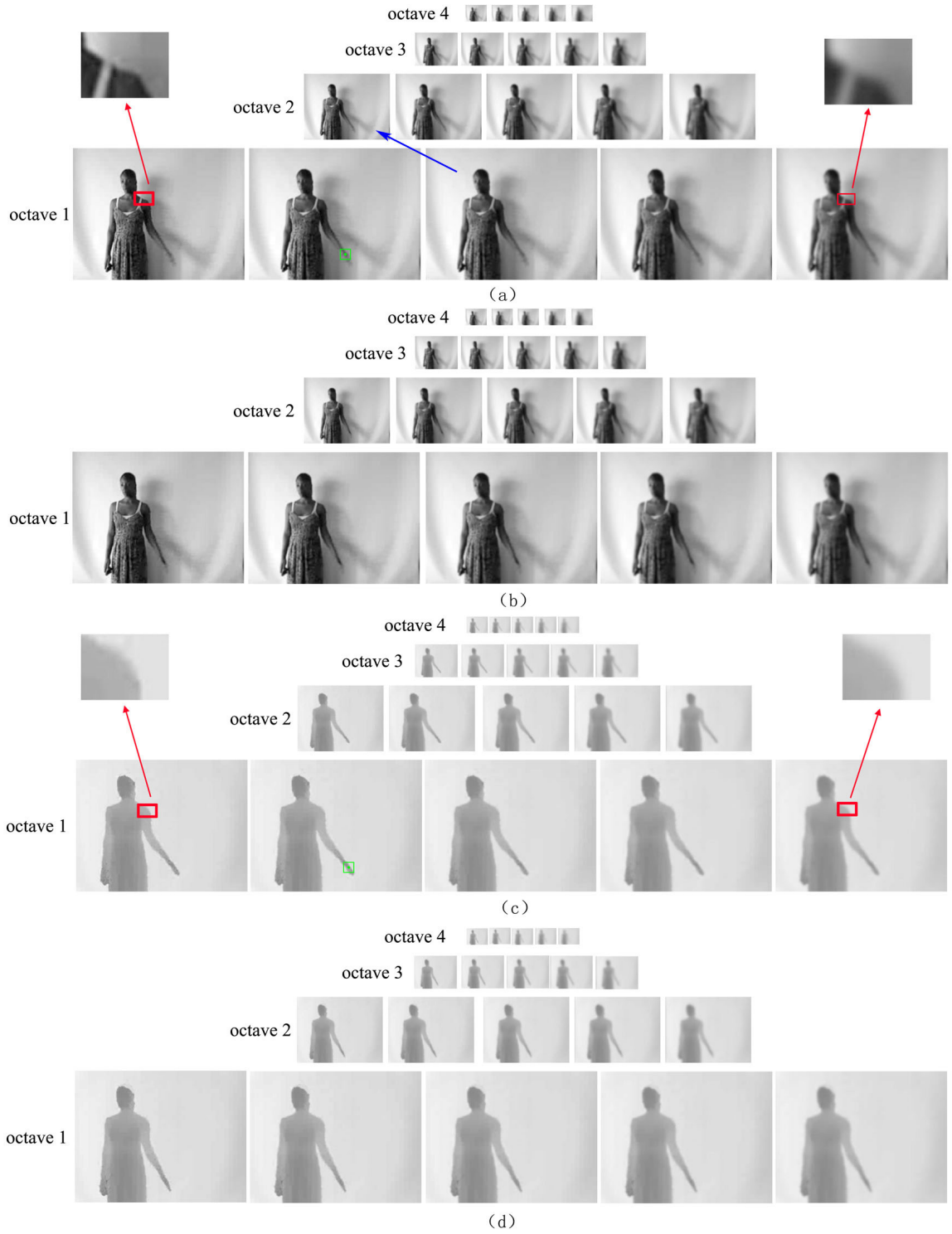


Figure 3: Building Gaussian pyramids and depth Gaussian pyramids for two consecutive frames. (a) the gaussian pyramid L^{I_t} at time t; (b) the gaussian pyramid $L^{I_{t+1}}$ at time $t+1$; (c) the depth gaussian pyramid L^{D_t} at time t; (d) the depth gaussian pyramid $L^{D_{t+1}}$ at time $t+1$.

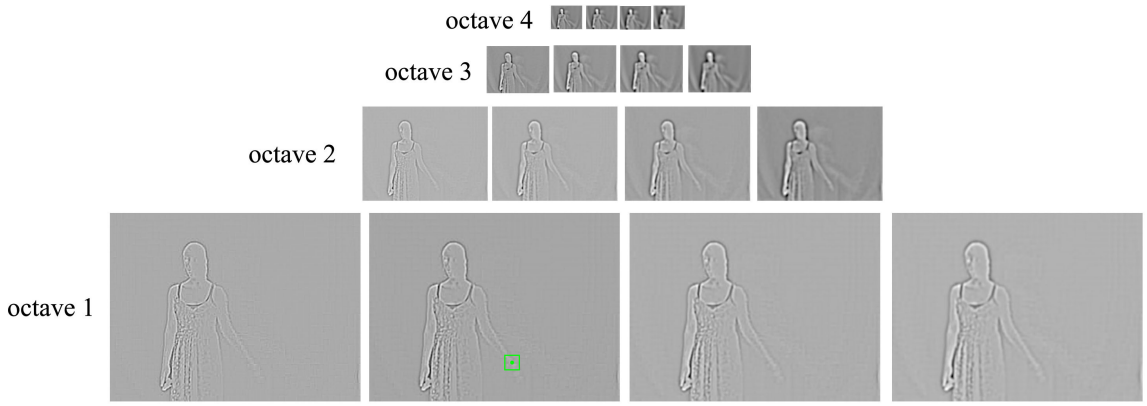


Figure 4: Building the difference of Gaussian pyramid Df^{I_t} from Figure 3(a) at time t.

the optical flow vector (v^{ρ}) of the point ρ can be solved by the least squares principle (Lucas et al., 1981). Namely, it solves:

$$Av^{\rho} = b,$$

where $A = \begin{bmatrix} F1_x(q_1) & F1_y(q_1) \\ F1_x(q_2) & F1_y(q_2) \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ F1_x(q_n) & F1_y(q_n) \end{bmatrix}$, $v^{\rho} = \begin{bmatrix} v_x^{\rho} \\ v_y^{\rho} \end{bmatrix}$, and $b = \begin{bmatrix} -F1_t(q_1) \\ -F1_t(q_2) \\ \cdot \\ \cdot \\ \cdot \\ -F1_t(q_n) \end{bmatrix}$, $q_1, q_2, ..., q_n$ are the pixels inside the window around the point ρ, $F1_x(q_i)$ and $F1_y(q_i)$ calculated by different operators (e.g., Scharr operator, Sobel operator) are the partial derivatives of the image $F1$ along the horizontal and vertical directions, and $F1_t(q_i) = F2(q_i) - F1(q_i)$ calculated by two consecutive frames is the partial derivatives along time. Besides, v_x^{ρ} (v_y^{ρ}) denotes the horizontal (vertical) velocity of the point ρ. So we can know the optical flow $V = [V_x\ V_y]^T$ of all the points in the image $F1$ via Equation (3).

$$[V_x\ V_y]^T = \bigcup_{i=1}^{\zeta} [v_x^{\rho_i}\ v_y^{\rho_i}]^T, \tag{3}$$

where ζ is the number of points in the image $F1$, $v_x^{\rho_i}$ ($v_y^{\rho_i}$) denotes the horizontal (vertical) velocity of the point ρ_i, and V_x (V_y) denotes the horizontal (vertical) component of the estimated optical flow for all the points in an image. In order to facilitate the following description, we rewrite Equation (3), so as to define $OpticalFlowKL(F1, F2)$, as follow:

$$[V_x\ V_y]^T = OpticalFlowKL(F1, F2) \stackrel{def}{=} \bigcup_{i=1}^{\zeta} [v_x^{\rho_i}\ v_y^{\rho_i}]^T.$$

Next, once two Gaussian pyramids (L^{I_t} and $L^{I_{t+1}}$) shown in Figure 3(a) and (b) are obtained at time t and $t+1$, respectively, we can calculate the optical flow at each interval of each octave via Equation (4). That is say,

$$[V^{I_t}_{x,(i,j)}\ V^{I_t}_{y,(i,j)}]^T = OpticalFlowKL(L^{I_t}_{i,j}, L^{I_{t+1}}_{i,j}),\ 0 \le i < n, 0 \le j < s+3, \tag{4}$$

where $L^{I_t}_{i,j}$ denotes the blurred image of the $(j+1)^{th}$ interval in the $(i+1)^{th}$ octave at time t, n and s are defined the same as Equation (1).

So the horizontal and vertical optical flow pyramids at time t are the union sets $\bigcup_{i,j} V^{I_t}_{x,(i,j)}$ and $\bigcup_{i,j} V^{I_t}_{y,(i,j)}$, respectively. For example, we use the Gaussian pyramids in Figure 3(a) and (b) to compute the optical flow pyramid via Equation (4). And the results are illustrated in Figure 5(a) and (b) where we can see that the highlighted parts occur around the motion parts.

Local Extrema Detection. Here, we describe three different methods (SIFT, 3D MoSIFT, 3D EMoSIFT) for interest point detection and show the similarities and differences among these methods.

(1) Local Extrema Detection: SIFT

In order to detect the local maxima and minima in the DoG pyramid $Df^{I_t}_{i,j}$, each point is compared to its eight neighbors in the current image and nine neighbors in the above and below images of each octave, which is illustrated in Figure 6(a). A point is selected only if it is larger than all of these neighbors or smaller than all of them. In Figure 4, the DoG pyramid $Df^{I_t}_{i,j}$ has four octaves and each octave has four images at time t. So we can find the local extrema points in the middle of two images at each octave, namely, $Df^{I_t}_{i,j}$, $\forall i \in [0,3], j \in [1,2]$. For example, in the first octave, we detect the local extrema points at the second image $Df^{I_t}_{0,1}$ (via comparing the point to his 8 neighbor points in the current image $Df^{I_t}_{0,1}$, 9 neighbor points in the image $Df^{I_t}_{0,0}$, and 9 neighbor points in the image $Df^{I_t}_{0,2}$) and the third image $Df^{I_t}_{0,2}$ (via comparing the point to his 8 neighbor points in the current image $Df^{I_t}_{0,2}$, 9 neighbor points in the image $Df^{I_t}_{0,1}$, and 9 neighbor points in the image $Df^{I_t}_{0,3}$). So we can detect the local extrema points in other octaves similar to the first octave. The detected points (marked with red points) are shown in Figure 7(a), which shows that many redundant points are detected in the background and torso regions.

(2) Local Extrema Detection: 3D MoSIFT[2]

3D MoSIFT first detect the local extrema like SIFT algorithm. Then those local extrema can only become interest points when those points have sufficient motion in the optical flow pyramid. That is say, if a point is treated as an interest point, the velocity of this point should satisfy the following condition:

$$v_x \ge \beta_1 \times w, v_y \ge \beta_1 \times h, \tag{5}$$

where v_x (v_y) is the horizontal (vertical) velocity of a point from the horizontal (vertical) optical flow pyramid V_x (V_y); β_1 is a pre-defined threshold; w and h are the width and height of the blurred image in the scale space.

As shown in Figure 5(a) and (b), we can see that only the local extrema located in the highlighted parts of the optical flow pyramids ($V^{I_t}_x$ and $V^{I_t}_y$) will become interest points. Because only the points in the highlighted parts have large motions, which may satisfy the condition in Equation (5). Other extrema will be eliminated, because they have no sufficient motion in the optical flow pyramids. The final results (marked with red points) are shown in Figure 7(b).[3] Comparing with SIFT algorithm, we can see that if the points are still, they will be filtered out via the conditions in Equation (5).

2. MoSIFT and 3*D* MoSIFT have the same strategy to detect interest points.
3. Here, $\beta_1 = 0.005$ according to the reference (Ming et al., 2012).

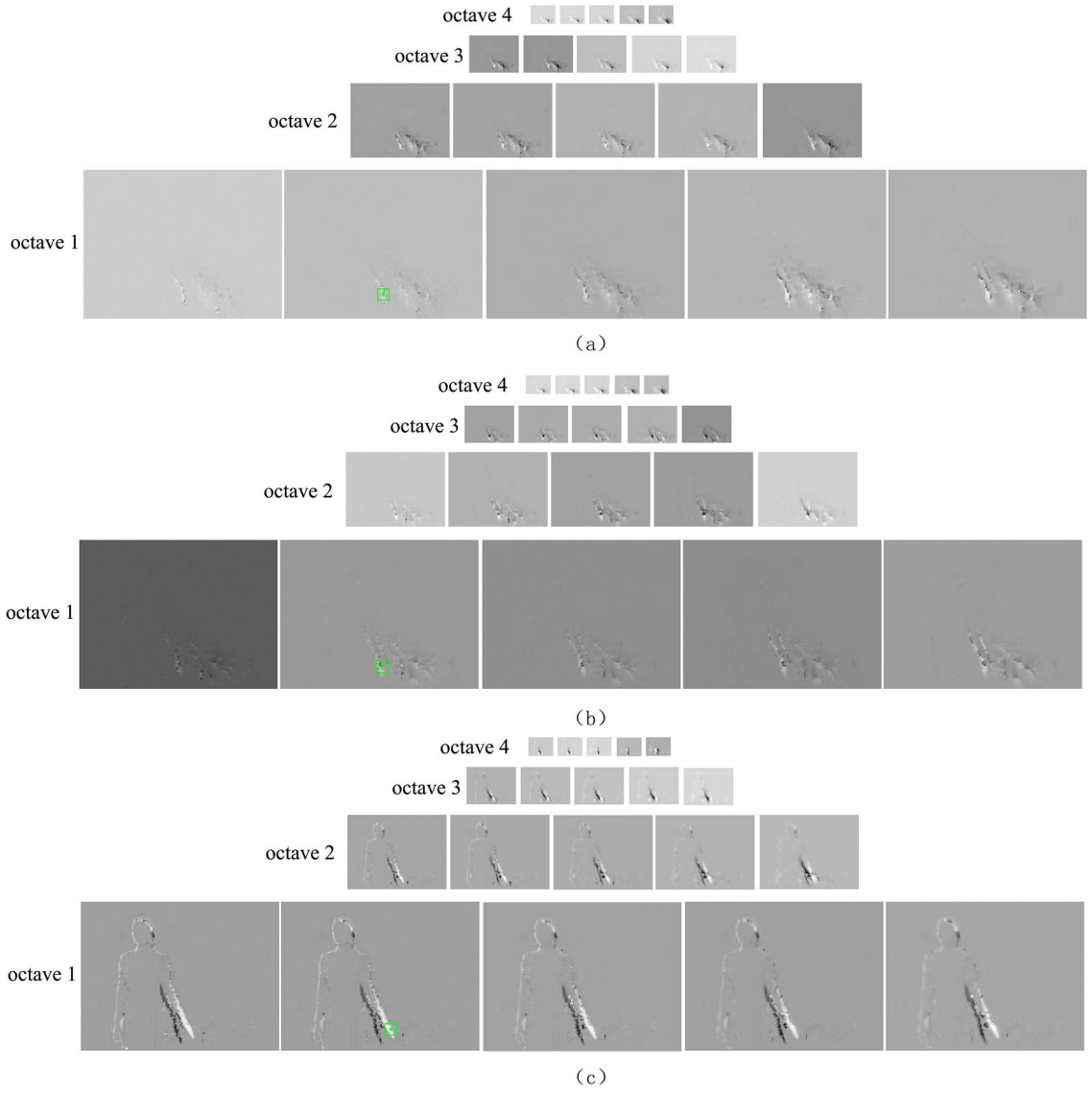


Figure 5: The horizontal and vertical optical flow pyramids are calculated from Figure 3(a) and (b). (a) The horizontal component of the estimated optical flow pyramid $V_x^{I_t}$ at time t; (b) The vertical component of the estimated optical flow pyramid $V_y^{I_t}$ at time t; (c) The depth changing component $V_z^{D_t}$ at time t.

However, in Figure 7(b), some useless points (from the background and torso regions) are still detected, which indicate that 3D MoSIFT is sensitive to the slight motion.

(3) Local Extrema Detection: 3D EMoSIFT

To eliminate the effect of the slight motion, we introduce a new condition to filter out the detected points by the SIFT algorithm. According to the above mentioned description, we have ob-

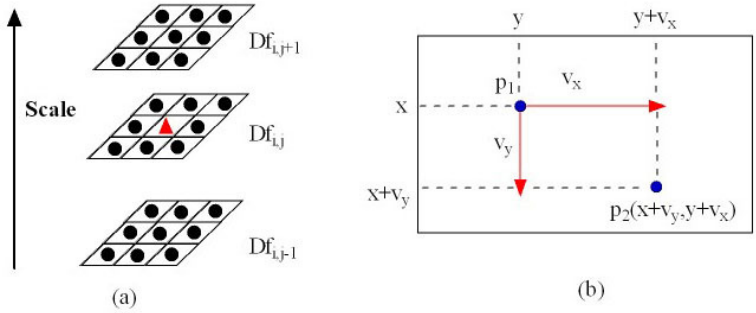


Figure 6: (a) The SIFT algorithm for interest points detection. Maxima and minima of the DoG images are detected by comparing a pixel (marked with a red triangle) to its 26 neighbors in 3×3 regions at the current and adjacent scales (marked with black circles); (b) the point prediction via the optical flow vector.

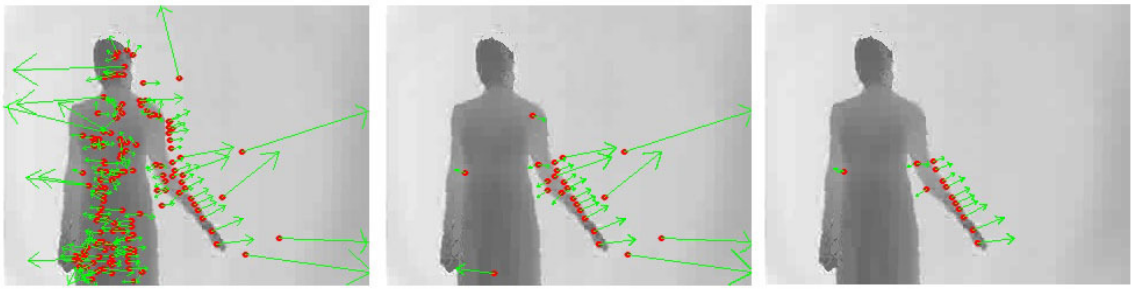


Figure 7: After interest point detection, the SIFT-based descriptors are calculated by three methods: SIFT, 3D MoSIFT and 3D EMoSIFT. The detected points are marked with red circles and the green arrows show the direction of movements. The figure shows that SIFT and 3D MoSIFT detect many useless points in the background and torso regions while the result by 3D EMoSIFT is more accurate. (a) SIFT; (b) 3D MoSIFT; (c) 3D EMoSIFT.

tained the pyramids $L^{D_t}, L^{D_{t+1}}, V_x^{I_t}, V_y^{I_t}$. For a given point p_1 from an image in different scale spaces at time t, we can easily know the horizontal and vertical velocities v_x, v_y by the corresponding image of the pyramids $V_x^{I_t}, V_y^{I_t}$. Then the predicted point p_2 at time $t+1$ can be calculated by the point p_1 at time t according to Figure 6(b). Therefore, we can know the depth changing component at time t as:

$$V_{z,(i,j)}^{D_t}(p_1) = L_{i,j}^{D_{t+1}}(p_2) - L_{i,j}^{D_t}(p_1),\ 0 \le i < n, 0 \le j < s+3. \tag{6}$$

Figure 5(c) shows the depth changing pyramid via Equation (6). We can see that the highlighted parts accurately occur in the gesture motion region. Therefore, the local extrema shown in Figure 7(a) by SIFT algorithm will become interest points when those points not only have sufficient motion which is satisfied with the condition of 3D MoSIFT in Equation (5) but also have enough depth

changing which is shown in the highlighted regions of Figure 5(c). That is say, the interest point detection must simultaneously satisfy the condition in Equation (5) and a new condition defined as:

$$v_z \geq \beta_2 \times \sqrt{w^2 + h^2}, \tag{7}$$

where v_z is the depth changing value of a point from the depth changing pyramid V_z; β_2 is a pre-defined threshold. The final results is shown in Figure 7(c).[4] We can see that 3D EMoSIFT can filter out the still points and the points with slight motion.

3.1.2 FEATURE DESCRIPTORS

The previous operations assigned an image location and scale to each interest point. That is say we can use the interest point to select the Gaussian images from different pyramids. Here, we give an example to illustrate how to compute the feature descriptor vector which is similar to the process in Ming et al. (2012). We assume that a detected point (marked with green dot) is found in DoG pyramid $Df_{0,1}^{I_t}$ at time t in Figure 4, which indicates that the detected point locates at the second image of the first octave. Then the corresponding points (marked with green dot) in different pyramids are shown in Figure 3 and Figure 5 at time t. To calculate the feature descriptors, we first extract the local patches ($\Gamma_1 \sim \Gamma_5$) around the detected point in five pyramids ($L^{I_t}, L^{D_t}, V_x^{I_t}, V_y^{I_t}$ and $V_z^{I_t}$), where Γ_1 is extracted from $L_{0,1}^{I_t}$, Γ_2 from $L_{0,1}^{D_t}$, Γ_3 from $V_{x,(0,1)}^{I_t}$, Γ_4 from $V_{y,(0,1)}^{I_t}$ and Γ_5 from $V_{z,(0,1)}^{D_t}$. These five patches are labeled as green rectangles in Figure 3 and Figure 5. The local patches $\Gamma_1 \sim \Gamma_5$ are of the same size 16×16 pixels and are shown in Figure 8. We first consider the appearance properties to construct the 3D gradient space via local patches Γ_1 and Γ_2. Then we use the rest of local patches (Γ_3, Γ_4 and Γ_5) to construct 3D motion space.

Feature Descriptors in 3D Gradient Space. For a given point p with its coordinate (i, j), we can simply calculate its horizontal and vertical gradients from RGB-D data (Γ_1 and Γ_2) as follow:

$$\begin{aligned} I_x(i,j) &= \Gamma_1(i, j+1) - \Gamma_1(i,j), \\ I_y(i,j) &= \Gamma_1(i+1, j) - \Gamma_1(i,j), \\ D_z^x(i,j) &= \Gamma_2(i, j+1) - \Gamma_2(i,j), \\ D_z^y(i,j) &= \Gamma_2(i+1, j) - \Gamma_2(i,j), \end{aligned}$$

where $I_x(i,j)$ and $I_y(i,j)$ are the horizontal and vertical gradients calculated from Γ_1; D_z^x and $D_z^y(i,j)$ are the horizontal and vertical gradients from Γ_2. We can calculate four gradients (I_x, I_y, D_z^x and D_z^y) for each point. Because the local patches (Γ_1 and Γ_2) are of size 16×16, there are 256 points and each point has four gradient values.

Then, as shown in Figure 8(a), for each point p, the 3D gradient space can be constructed by $I_x(i,j), I_y(i,j), D_z^x(i,j)$ and $D_z^y(i,j)$. Now we use the xy plane to illustrate how to calculate the feature descriptor in the 3D gradient space. For each point p with its coordinate (i, j), we compute the gradient magnitude, $mag(i,j) = \sqrt{I_x(i,j)^2 + I_y(i,j)^2}$, and orientation, $ori(i,j) = tan^{-1}(I_y(i,j)/I_x(i,j))$ in the xy plane. Then, in xy plane, we can generate a new patch Γ_{xy} which is the left image in the first row of Figure 8(c). The size of Γ_{xy} is 16×16. For each point with its coordinate (i, j) from Γ_{xy}, it has two values: the gradient magnitude $mag(i,j)$ and orientation $ori(i,j)$. Γ_{xy} can be divided into

4. Here, $\beta_1 = \beta_2 = 0.005$.

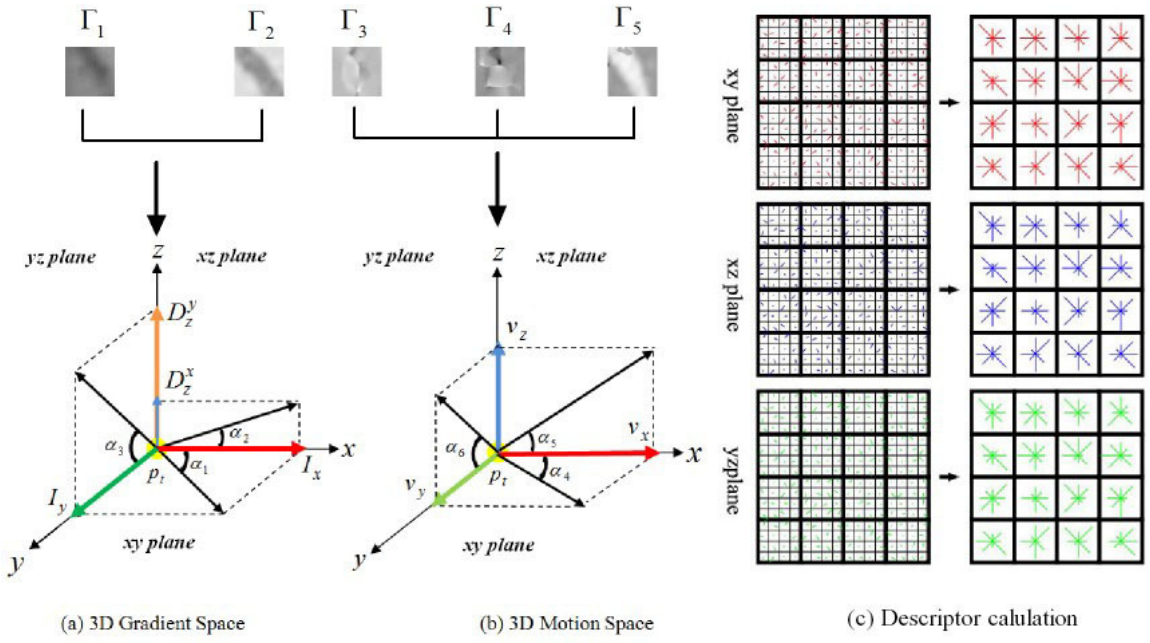


Figure 8: Computing the feature descriptor in two parts: (a) 3D Gradient Space, (b) 3D Motion Space, (c) Feature descriptor calculation

16 (4×4) grids. For each grid with 4×4 points, we calculate its orientation histogram with 8 bins, which means the orientation is grouped into 8 directions which is represented by the right image in the first row of Figure 8(c). This leads to a descriptor vector with 128 ($4 \times 4 \times 8$) dimensions in xy plane. Here, each sample added to the histogram is weighed by its gradient magnitude and by a Gaussian weighting function (Lowe, 2004). Similarly, we can calculate the descriptors in xz and yz planes. Therefore, the descriptor vector of the 3D gradient space has 384 (128×3) dimensions.

Feature Descriptors in 3D Motion Space. For a given point p with coordinates $(i, j), \forall\ 0 \leq i \leq 15,\ 0 \leq j \leq 15$, we can easily know the velocities according to the local patches Γ_3, Γ_4, and Γ_5. That is say, $v_x(i, j) = \Gamma_3(i, j)$, $v_y(i, j) = \Gamma_4(i, j)$ and $v_z(i, j) = \Gamma_5(i, j)$.

Thus, we can construct the 3D motion space as shown in Figure 8(b). Similar to the descriptor calculation in 3D gradient space, we can compute the magnitude and orientation (using v_x, v_y, v_z) for the local patch around the detected points in three planes. The only difference is that v_z is the same in both xz and yz planes. Therefore, we obtain the descriptors with 384 dimensions in the 3D motion space. Finally, we integrate these two descriptor vectors into a long descriptor vector with 768 dimensions.

3.1.3 OVERVIEW THE 3D EMOSIFT FEATURES

In this section, we propose a new spatio-temporal feature called 3D EMoSIFT. Each 3D EMoSIFT feature descriptor has 768 dimensions. Since the 3D EMoSIFT feature is derived from SIFT algorithm, the features are invariant to scale and rotation. Besides, compared to other similar features (SIFT, MoSIFT, 3D MoSIFT), the new features can capture more compact motion patterns and are

not sensitive to the slight motion (see the Figure 7). For a given sample including an RGB video and a depth video, we can calculate feature descriptors between two consecutive frames. Then the sample can be represented by the set of all the feature descriptors extracted from the video clips. Algorithm 1 illustrates how to calculate the proposed features.

Now each sample is denoted by the set of descriptor vectors, and we want to use those vectors for BoF representation. To do that, we will create histograms counting how many times a descriptor vector (representing a feature) appears at interest points anywhere in the video clip representing the gesture. There is a need to first replace the descriptor vectors by codes to limit the number of features, otherwise there would be too many entries in the histogram and the representation would be too sparse. So, we will describe the means of creating a codebook in the next Section 3.2.

Algorithm 1 The algorithm for the 3D EMoSIFT feature

Input:

- A sample with two videos: $V_r = [I_1, I_2, ..., I_Q]$ (RGB data), $V_d = [D_1, D_2, ..., D_Q]$ (depth data)
- Number of frames : Q

Output:

- The set of feature descriptors : X

Initialization: $X = [\,]$
for $i = 1$ to $Q-1$ **do**
  Obtain the frames: I_i and I_{i+1} from V_r; D_i and D_{i+1} from V_d
  Build the Gaussian Pyramids: $L^{I_i}, L^{I_{i+1}}, L^{D_i}$ and $L^{D_{i+1}}$ via Equation (1)
  Build the different of Gaussian (DoG) Pyramid: Df^{I_i} via Equation (2)
  Build the Optical Flow Pyramids: $V_x^{I_i}$ and $V_y^{I_i}$ via Equation (4)
  Build the depth changing Pyramid: V_z^{Di} via Equation (6)
  Find the set of interest points: $P = [p_1, ..., p_m]$ via Figure 6(a), Equation (5) and (7)
  for $j = 1$ to m **do**
    Get the information of the interest point from the set P: p_i
    Compute feature descriptor from the local patch around p_i: $x \in \Re^{768}$ via Figure 8
    $X = [X\ x]$
  end for
end for
return X

3.2 Codebook Learning and Coding Descriptors

Suppose the matrix X is the set of all descriptor vectors for an entire video clip representing a gesture, and $X = [x_1, x_2, ..., x_N] \in \Re^{d \times N}$, where x_i denotes a description with d dimensions. A codebook B with M entries is denoted with $B = [b_1, b_2, ..., b_M] \in \Re^{d \times M}$. The coding methods map each descriptor into a M-dimensional code to generate the video representation. We first introduce how to learn a codebook B, then review VQ and introduce SOMP for code descriptors.

3.2.1 CODEBOOK LEARNING

Let η denote the number of gesture classes (that means there are η training samples for one-shot learning), $\Omega = [X^1, X^2, ..., X^\eta]$, $\Omega \in \Re^{d \times L_{tr}}$ is the set of all the descriptor vectors extracted from all the training samples, $X^i \in \Re^{d \times N_i}$ with N_i descriptor vectors is the set extracted from the i^{th} class, and $L_{tr} = \sum_{i=1}^{\eta} N_i$ is the number of features extracted from all the training samples. Then we learn the codebook $B \in \Re^{d \times M}$ ($M < \sum_{i=1}^{\eta} N_i$) with M entries by applying the k-means algorithm (Wang et al., 2010) over all the descriptors Ω in our work. However, unlike traditional BoF models, we use a new parameter $\gamma \in (0,1)$ instead of the codebook size M (The way we select γ will be discussed in Section 4.). γ is expressed as a fraction of L_{tr}. Therefore, the codebook size M can be calculated below:

$$M = L_{tr} \times \gamma. \tag{8}$$

3.2.2 CODING DESCRIPTORS BY VQ

In the traditional VQ method, we can calculate the Euclidean distance between a given descriptor $x \in \Re^d$ and every codeword $b_i \in \Re^d$ of the codebook B and find the closest codeword. The VQ method can be formulated as:

$$\min_C \|X - BC\|_F^2, \ s.t. \|c_i\|_0 = 1, \|c_i\|_1 = 1, c_i \geqslant 0, \ \forall \ i, \tag{9}$$

where $\|\cdot\|_F$ is the Frobenius norm, $C = [c_1, c_2, ..., c_N] \in \Re^{M \times N}$ is the set of codes for X, $\|\cdot\|_0$ is the ℓ_0 norm that counts the number of nonzero elements, $\|\cdot\|_1$ is the ℓ_1 norm; The conditions $\|c_i\|_0 = 1, \|c_i\|_1 = 1, c_i \geqslant 0$, mean that only one element is equal to 1 and the others are zero in each code $c_i \in \Re^M$.

This formulation in Equation (9) allows us to compare more easily with sparse coding (see the Section 3.2.3). In Equation (9), the conditions may be too restrictive, which gives rise to usually a coarse reconstruction of X. Therefore, we use a sparse coding method instead of VQ.

3.2.3 CODING DESCRIPTORS BY SOMP

Inspired by image classification (Yang et al., 2009) and robust face recognition (Wright et al., 2009) via sparse coding, we relax the restricted conditions in Equation (9) and suppose X has a sparse representation $C = [c_1, c_2, ..., c_N]$, $c_i \in \Re^M$ that means each c_i contains k ($k \ll M$) or fewer nonzero elements. Then, the problem can be stated as the following optimization problem:

$$\min_C \|X - BC\|_F^2, \ s.t. \|c_i\|_0 \leq k, \ \forall \ i. \tag{10}$$

Solving Equation (10) accurately is an NP-hard problem (Wright et al., 2009; Guo et al., 2013). Nevertheless, approximate solutions are provided by greedy algorithms or convex relaxation, such as SOMP (Tropp et al., 2006; Rakotomamonjy, 2011). To the best of our knowledge, we are the first to use SOMP in BoF model for gesture recognition, especially for one-shot learning gesture recognition.

Then we give a brief introduction about the SOMP algorithm and analyze the computational complexity. SOMP is a greedy algorithm which is based on the idea of selecting an element of the codebook and building all signal approximations as the projection of the signal matrix X on the span

of these selected codewords. This algorithm (Tropp et al., 2006; Rakotomamonjy, 2011) is shown in Algorithm 2. Regarding the computational complexity, we note that the most demanding part of the SOMP is the correlation E computation which has the complexity $O(dMN)$. And the complexity of the linear system to be solved for obtaining C at each iteration is $O(|\Lambda|)$. So the complexity for k iterations is about $O(dkMN) + O(k|\Lambda|)$. Although the complexity of SOMP is more expensive than VQ which has $O(dMN)$ (Linde et al., 1980). SOMP has several merits which will be discussed later.

Algorithm 2 The SOMP algorithm

Input:

- A signal matrix (the feature set): $X = [x_1, x_2, ..., x_N] \in \Re^{d \times N}$
- A learned codebook: $B = [b_1, b_2, ..., b_M] \in \Re^{d \times M}$
- the sparsity: k

Output:

- The sparse representation: C

Initialization: the residual matrix $R_s = X$, the index set $\Lambda = [\,]$;
for $i = 1$ to k **do**
  $E = B^T R_s$, where $E = \cup_{p,q}[e_{p,q}]$
  Find the index $\lambda = argmax_q \sum_p |e_{p,q}|$
  $\Lambda = [\Lambda\ \lambda]$
  $C = (B_\Lambda^T B_\Lambda)^{-1} B_\Lambda^T X$
  $R_s = X - BC$
end for
return C

When the codebook $B \in \Re^{d \times M}$ and a descriptor set $X \in \Re^{d \times N}$ are given, the set of codes $C \in \Re^{M \times N}$ can be calculated by the coding methods (VQ or SOMP). Then the mean reconstruction error (MRE) for X is defined as:

$$\varepsilon_{MRE} = \sum_{i=1}^{N} \varepsilon_i / N,$$

where $\varepsilon_i = \|x_i - Bc_i\|_2^2$ is the reconstruction error of the i^{th} descriptor.

To compare the $MREs$ for both the VQ and SOMP methods, a matrix $X \in \Re^{64 \times 2000}$ is randomly generated based on the standard normal distribution. Then the matrix X is split into two parts ($X_1 \in \Re^{64 \times 1000}$ and $X_2 \in \Re^{64 \times 1000}$). The matrix X_1 is used to build a codebook B by the k-means algorithm. Then we use X_2 to calculate the codes C_{VQ} and C_{SOMP} via Equation (9) and (10), respectively. Finally we calculate the $MREs$ under varied cluster numbers and different sparsity values $k = \{5, 10, 15\}$. Figure 9 shows the results of both coding methods. We can see that the $MREs$ of the SOMP method is much lower than the $MREs$ of the VQ method.

Compared with the VQ method, SOMP has several advantages. First, the codebook B is usually overcomplete (i.e., $M > d$). Overcomplete codings smoothly interpolate between input vectors and are robust under input noise (Olshausen et al., 1997). Second, SOMP achieves a much lower reconstruction error. Although there is no direct relationship between lower reconstruction error and good recognition results, some authors (Yang et al., 2009; Wan et al., 2012) have shown that oftentimes better reconstruction leads to better performance. Third, the sparsity prior allows the

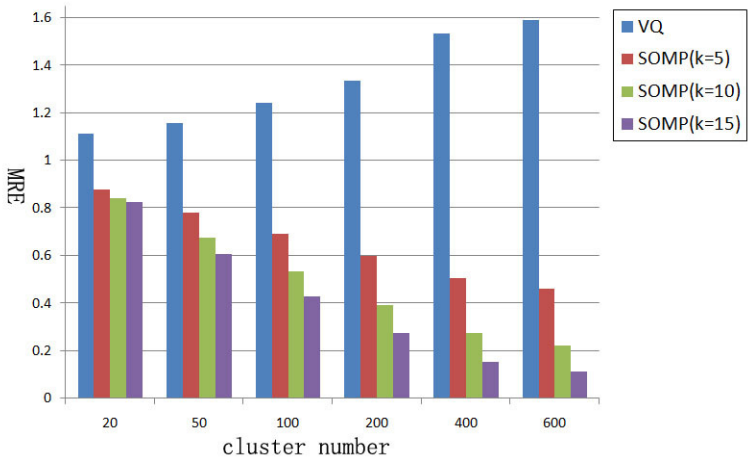


Figure 9: Comparison *MREs* using both VQ and SOMP methods.

learned representation to capture salient patterns of local descriptors. According to our experimental results in Section 4, although VQ can produce satisfactory accuracy, SOMP can achieve better performance.

3.3 Coefficient Histogram Calculation and Classification

The matrix X contains the descriptors obtained from a test sample and C contains their corresponding sparse representations over the learned codebook B. The sparse coefficients of the vector $c_i \in C$ present the contribution of all the entries in approximating the descriptor $x_i \in X$. The sparse coefficients associated with all the descriptors of the test sample thus collectively demonstrate the contribution of the entries toward the representation of that sample. Therefore, we use the coefficient histogram to denote the representation of each individual sample via Equation (11).

$$h = \frac{1}{N}\sum_{i=1}^{N} c_i, \tag{11}$$

where $c_i \in \Re^M$ is the i^{th} descriptor of $C \in \Re^{M\times N}$, and N is the total number of descriptors extracted from a sample and $h \in \Re^M$.

Because we have only one sample per class for training, multi-class SVMs are not trivially applicable because they require in principle a large number of training examples. So we select the NN classification for gesture recognition.

In the above discussion, we assume that every video has one gesture but this assumption is not suitable for continuous gesture recognition system. Therefore, we first apply DTW to achieve temporal gesture segmentation, which splits the multiple gestures to be recognized. We use the sample code about DTW provided in ChaLearn gesture challenge website (`http://gesture.chalearn.org/data/sample-code`). The detailed description of how to use DTW in one-shot learning can be found in Guyon et al. (2013). We briefly introduce the process for temporal gesture segmentation by DTW so as to make this paper more self-contained.

3.4 Temporal Gesture Segmentation based on DTW

Let $V = [I_1, ..., I_N]$ be a video with N frames, where I_i is the i^{th} frame (grayscale image) in the video. A video is represented by a set of motion features obtained from difference images as follows. First, the difference image is computed by subtracting consecutive frames in a video, that is $E_i = I_{i+1} - I_i$, $i = 1, ..., N-1$. The difference image is shown in Figure 10(b). Then a grid of equally spaced cells is defined over the difference image. The default size of the grid is 3×3 as shown in Figure 10(c). For each cell, we calculate the average value in the difference image, so a 3×3 matrix is generated. Finally, we flatten this matrix into a vector which is called motion feature. Therefore, a video V with N frames is represented by a matrix (the set of motion features) $f_V \in \Re^{9 \times (N-1)}$.

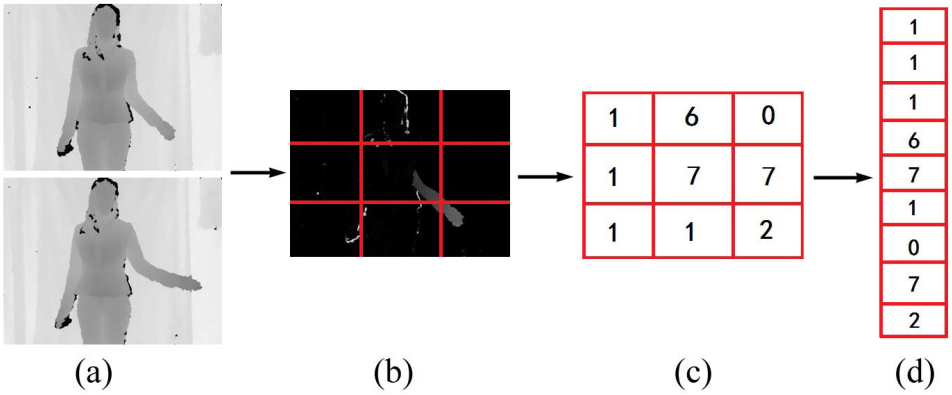


Figure 10: An example for the calculation of motion feature vector.

The reference sequence with κ training videos is denoted by $F_{tr} = [f_{Vtr_1}, ..., f_{Vtr_\kappa}]$, f_{Vtr} is the set of motion features of a training video. A test sequence is denoted by $F_{te} = f_{Vte}$ (the set of motion features for the test video). We calculate the negative Euclidean distance between each entry (a motion feature) from F_{tr} and each entry (a motion feature) from F_{te}. Then we calculate the DTW distance and apply the Viterbi algorithm (Viterbi, 1967) to find the temporal segmentation (an optimal warping path). In Figure 11, the left gray image shows the set of motion features (F_{tr}) as the reference sequence calculated from training videos. A motion feature (F_{te}) as the test sequence is computed from a new input video. The optimal path is shown in the top right corner (the green line is the optimal path; the short red lines are the boundary of two neighboring gestures). We can see that the testing video is splitted into five gestures.

3.5 Overview of the Proposed Approach

In this section, we describe the proposed approach based on bag of 3D EMoSIFT features for one-shot learning gesture recognition in detail. In the recognition stage, it has five steps: temporal gesture segmentation by DTW, feature descriptor extraction using 3D EMoSIFT, coding descriptor via SOMP, coefficient histogram calculation and the recognition results via NN classifier. The overall process is summarized in Algorithm 3.

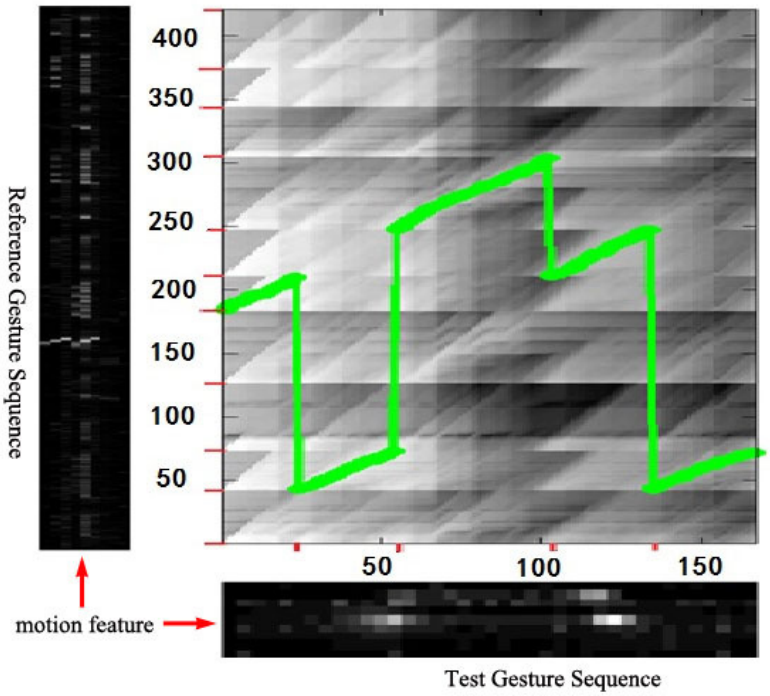


Figure 11: Temporal gesture segmentation by DTW.

4. Experimental Results

This section summarizes our results and demonstrates the proposed method is well suitable for one-shot learning gesture recognition. We first discuss the parameters of the proposed method. We further extend our method to compare with other state-of-the-art methods. Our experiments reveal that the proposed method gives superior recognition performance than many existing approaches.

4.1 Database

We evaluate the proposed method on development batches ($devel01 \sim devel20$), validation batches ($valid01 \sim valid20$) and final batches ($final21 \sim final40$) which contain in total 6000 gestures. The sixty batches are from Chalearn gesture challenge. Each batch is made of 47 gesture videos and split into a training set and a test set. The training set includes a small set of vocabulary spanning from 8 to 15 gestures. Every test video contains 1 to 5 gestures. Detailed descriptions of the gesture data can be found in Guyon et al. (2012). All the samples are recorded with a Microsoft $Kinect^{TM}$ camera which provides both RGB and depth video clips. Some examples are shown in Figure 12 where the first row is RGB images and the corresponding depth images are shown in the second row.

4.2 Metric of Evaluation

We adopt the metric of evaluation that was used by the challenge organizers (Guyon et al., 2012) to rank the entries. To evaluate performance, we use Levenshtein distance to calculate the score between the predicted labels and the truth labels. This distance between two strings is defined as

Algorithm 3 The proposed approach for one-shot learning gesture recognition

The condition for one-shot learning: given K training samples (RGB-D data) for K class (one sample per gesture class).

Input:

- Training samples (RGB-D data): $T_r = [t_{r1}, ..., t_{rK}]$
- A learned codebook: B (computed from training stage)
- Coefficient histograms of training samples: $H_r = [h_{r1}, h_{r2}, ..., h_{rK}]$ via Equation (11) (computed from training stage)
- A test sample (RGB-D data): t_e

Output:

- The recognition results: $class$

Initialization: $class = [\,]$
Temporal gesture segmentation: $[t_{e_1}, t_{e_2}, ..., t_{e_N}] = DTW(T_r, t_e), N \geq 1$
for $i = 1$ to N **do**
  Spatio-temporal feature extraction: $X_{t_e} = 3D\_EMoSIFT(t_{e_i})$
  For X_{t_e}, calculate its sparse representation C over the pre-trained codebook B
  $\min_C \|X_{t_e} - BC\|_F^2 \quad s.t. \quad \|c_j\|_0 \leq k, \forall\ j$
  Calculate the coefficient histogram h_{t_e} via Equation (11)
  Recognition: $tmp\_calss = nn\_classify(H_r, h_{t_e})$
  $class = [class\ tmp\_calss]$
end for
return $class$

Figure 12: Some samples from ChaLearn gesture database.

the minimum number of operations (insertions, substitutions or deletions) needed to transform one string into the other. In our case, the strings contain the gesture labels detected in each sample. For all comparisons, we compute the mean Levenshtein distance (MLD) over all video clips and batches. MLD score is analogous to an error rate (although it can exceed 1).

4.3 Parameters Discussion

This part gives the discussion of the parameters of the proposed method. First, we analysis the parameters of 3D EMoSIFT. Then, two parameters from the BoF model are discussed.

4.3.1 PARAMETERS OF 3D EMOSIFT

There are five parameters for constructing 3D EMoSIFT features. Three parameters σ, n and s in Equation (1) are derived from SIFT algorithm. We set $\sigma = 1.6$ and $s = 3$. Because Lowe (2004) suggest that when $\sigma = 1.6$ and $s = 3$, they can provide the optimal repeatability according to their experimental results. Besides, the number of octaves n can be calculated according to the original image size, such as $int(log_2(min(width, height)))$ (Vedaldi and Fulkerson, 2008).

The rest of parameters are β_1 in Equation (5) and β_2 in Equation (7). β_1 and β_2 determine the detection of interest points based on motion and depth change. When β_1 and β_2 are smaller, more interest points will be detected. We find that when $\beta_1 \in [0.003\ 0.008], \beta_2 \in [0.003\ 0.008]$, the performances are very stable as shown in Figure 13 where the results are calculated from two batches. We can see that MLD scores vary from 0.075 to 0.092 for *devel*01 batch, from 0.089 to 0.134 for *devel*02 batch. Therefore, $\beta_1 = \beta_2 = 0.005$ is used throughout this paper based on empirical results.

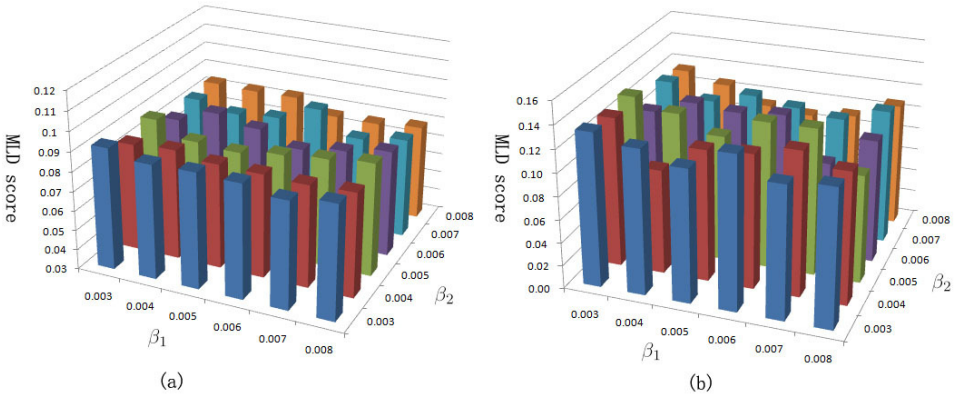


Figure 13: Parameters: $\sigma = 1.6$, $s = 3$, $\gamma = 0.2$ and $k = 10$. The MLD scores are calculated with different values β_1, β_2. (a) on *devel*01 batch (b) on *devel*02 batch

4.3.2 PARAMETERS OF THE BOF MODEL

There are two parameters in the BoF model: γ in Equation (8) and k in Equation (10) . Unlike traditional BoF models, we use a new parameter $\gamma \in (0, 1)$ to replace the codebook size M mentioned in Section 3.2. We first explain the reasons for choosing γ. Table 2 shows some information on different batches ($final21 \sim final40$), such as the number of training samples and the number of features extracted from training samples. We can see that the number of features varies on different batches. If a given codebook size M is too large, it may cause over-clustering on some batches where the number of features is relatively fewer (e.g., $final25$ and $final36$). Therefore, the over-clustering will effect the final MLD score. For instance, we evaluate seven different codebook sizes: {800, 1000, 1500, 2000, 2500, 3000, 3500}. The corresponding results are shown in Table 3 where the best performance is 0.18242. Then we evaluate different values {0.1, 0.2, 0.3} for γ, and the results are shown in Table 4. We can see that even though $\gamma = 0.1$, the corresponding MLD score is 0.17415 which can easily beat the best performance in Table 3. Additionally, when $\gamma = 0.1$, the

corresponding mean codebook size 1440 is much smaller than the given codebook size 3500 which is from the best result in Table 3.

The theory of sparse coding and the codebook learning are in a developing stage and the problems for selecting optimal parameters (e.g., γ, sparsity k) are still open issues (Guha and Ward, 2012). In this paper, we use a simple strategy to decide these two parameters. At first, we keep $k = 10$ and set γ with different values (ranging from 0.1 to 0.5), then determine γ by the lowest MLD score. Figure 14(a) shows the results. It reveals when $\gamma = 0.5$, we can get a higher performance and the corresponding MLD score is 0.13145. Then we set different values of k with $\gamma = 0.5$ and the results are shown in Figure 14(b). We can see that MLD scores remain stable. When $\gamma = 0.5$ and $k = 12$, the proposed method gets the lowest MLD score (the corresponding value is 0.1259).

batch names	number of training samples:N_{tr}	number of features (3D MoSIFT):$L1_{tr}$	number of features (3D EMoSIFT):$L2_{tr}$	decrease in ratio:$1-\frac{L2_{tr}}{L1_{tr}}$
final21	10	18116	13183	27.23%
final22	11	19034	15957	16.17%
final23	12	11168	7900	29.26%
final24	9	10544	7147	32.22%
final25	11	8547	6180	27.69%
final26	9	9852	7675	22.10%
final27	10	29999	20606	31.31%
final28	11	16156	10947	32.24%
final29	8	30782	22692	26.28%
final30	10	20357	14580	28.38%
final31	12	22149	17091	22.84%
final32	9	12717	10817	14.94%
final33	9	42273	29034	31.32%
final34	8	24099	16011	33.56%
final35	8	39409	27013	31.45%
final36	9	9206	6914	24.90%
final37	8	22142	14181	35.95%
final38	11	26160	18785	28.19%
final39	10	16543	11322	31.56%
final40	12	11800	10128	14.17%
Average	9.85	20052.65	14408.15	**28.15%**

Table 2: This table shows some information for every batch. The last row reveals the average number. Although the average number of 3D EMoSIFT features has decreased by 28.15%, 3D EMoSIFT has a higher performance than 3D MoSIFT in our experimental results. Besides, compared 3D MoSIFT features, the process time of 3D EMoSIFT can be faster to build the cookbook.

4.4 Comparisons

In order to compare with other methods, we first use the standard BoF model to evaluate different spatio-temporal features. Then the performances of VQ and SOMP is given. Besides, we evaluate

codebook size M	800	1000	1500	2000	2500	3000	3500
MLD score	0.21448	0.21504	0.19514	0.18961	0.18684	0.18574	**0.18242**

Table 3: Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$ and $k = 10$ ($final21 \sim final40$). MLD scores with different codebook sizes M.

γ	0.1	0.2	0.3
MLD score	0.17415	0.14753	0.14032
Mean codebook size	1440	2881	4322

Table 4: Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$ and $k = 10$ ($final21 \sim final40$). MLD scores with different values for γ.

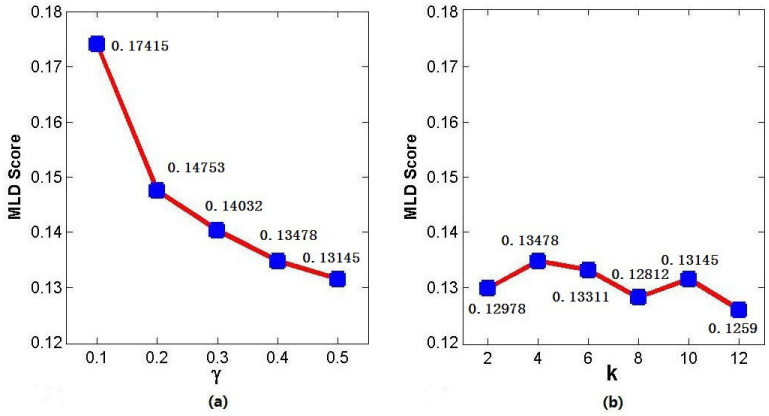


Figure 14: (a) Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$ and $k = 10$ ($final21 \sim final40$). MLD scores with different values of γ; (b) Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$ and $\gamma = 0.5$ ($final21 \sim final40$). MLD scores with different values of sparsity k.

the performances of both the gradient-based and motion-based features. Finally, we compare the proposed approach with some popular sequence matching methods.

4.4.1 COMPARISON WITH OTHER SPATIO-TEMPORAL FEATURES

In our experiments, we use the standard BoF model to evaluate different spatio-temporal features, which means VQ is used for coding descriptors. As shown in Figure 14(b), the results are relatively stable when sparsity k has different values. Therefore, we evaluate different values $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ for γ and set $k = 10$. The results are shown in Table 5, where we can draw the following conclusions.

First, the results of 3D EMoSIFT and 3D MoSIFT consistently exceed traditional features (e.g., Cuboid, Harris3D and MoSIFT). More specifically, the least MLD scores (corresponding to the best recognition rate) for 3D EMoSIFT is 0.13311, compared to 0.14476 for 3D MoSIFT, 0.28064 for Cuboid, 0.18192 for Harris3D, and 0.335 for MoSIFT.

Methods \ γ	0.1	0.2	0.3	0.4	0.5
Cuboid(R)	0.36717	0.36495	0.34332	0.33111	0.31392
Cuboid(R+D)	0.33666	0.31559	0.30948	0.30782	0.28064
Harris3D hog(R)	0.30061	0.26012	0.25014	0.23516	0.23461
Harris3D hog(R+D)	0.24903	0.22795	0.22407	0.22795	0.22684
Harris3D hof(R)	0.34831	0.32668	0.31281	0.29895	0.29063
Harris3D hof(R+D)	0.32169	0.29174	0.28508	0.27898	0.27121
Harris3D hoghof(R)	0.24237	0.21963	0.20022	0.19468	0.18857
Harris3D hoghof(R+D)	0.20965	0.18802	0.18303	0.18747	0.18192
MoSIFT(R)	0.41653	0.39601	0.35885	0.36606	0.33500
MoSIFT(R+D)	0.44426	0.44260	0.43594	0.42318	0.40488
3D MoSIFT(R+D)	0.19135	0.16694	0.16195	**0.14476**	**0.14642**
3D EMoSIFT(R+D)	0.16528	**0.15419**	**0.14753**	**0.13977**	**0.13311**

Table 5: Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$ and $k = 10$ ($final21 \sim final40$). It shows MLD scores by different spatio-temporal features with different values of γ, where (R) means the features are extracted from RGB video, (R+D) means the features are extracted from the RGB and depth videos. The values shown in bold indicate superior performance, with MLD scores below 0.16.

Second, from the previous works, we know that traditional features have achieved promising results (Dollár et al., 2005; Laptev, 2005; Chen and Hauptmann, 2009). However, those features may be not sufficient to capture the distinctive motion pattern only from RGB data because there is only one training sample per class.

Third, although 3D MoSIFT and 3D EMoSIFT are derived from the SIFT and MoSIFT features, MoSIFT still cannot achieve satisfactory outcomes. That is because the descriptors captured by MoSIFT are simply calculated from RGB data while 3D MoSIFT and 3D EMoSIFT construct $3D$ gradient and $3D$ motion space from the local patch around each interest point by fusing RGB-D data.

To show the distinctive views for both 3D MoSIFT and 3D EMoSIFT features, we record three gesture classes: clapping, pointing and waving. The samples are shown in Figure 15, where the training samples are shown in the first three rows (of the first two columns) and the testing samples are shown in the last three rows (of the first two columns). We first extract 3D MoSIFT and 3D EMoSIFT features from the six samples. Then we use 3D MoSIFT and 3D EMoSIFT features extracted from the three training samples to generate a codebook which has 20 visual words, respectively. Each descriptor is mapped into a certain visual word with VQ. The spatial distribution of visual words for each sample are shown in Figure 15 where different visual words are represented by different colors. It shows that 3D EMoSIFT is more compact. A more compact feature leads to a better performance (see Table 5) and can effectively reduce the redundant features (see Table 2). Besides, a compact feature should encourage the signals from the same class to have similar representations. In other words, the signals from the same class are described by similar histograms (or visual words). From the Figure 15, we can see that the samples from the same class have similar histograms (e.g., clapping gesture) when we use 3D EMoSIFT. However, 3D MoSIFT cannot

Figure 15: The first two columns are the samples used for training and testing. The third and fifth columns reveal the spatial distribution of the visual words for the samples, which show 3D EMoSIFT is more compact. We superimpose the interest points in all frames into one image. Different visual words are represented by different colors. The fourth and sixth columns are shown the histograms for each sample. The histogram vector is ℓ_2 normalization. It shows each class has some dominating visual words. A compact feature encourages gestures from the same class to be described by similar histograms (or visual words), especially the dominating visual words. The histograms from the same class learned by 3D EMoSIFT are similar (i.e., clapping gesture).

get good similar histograms. From the above discussions, we see that 3D EMoSIFT is suitable for one-shot learning gesture recognition. Interestingly, 3D EMoSIFT is also more sparsity than 3D MoSIFT (see the histograms in Figure 15)

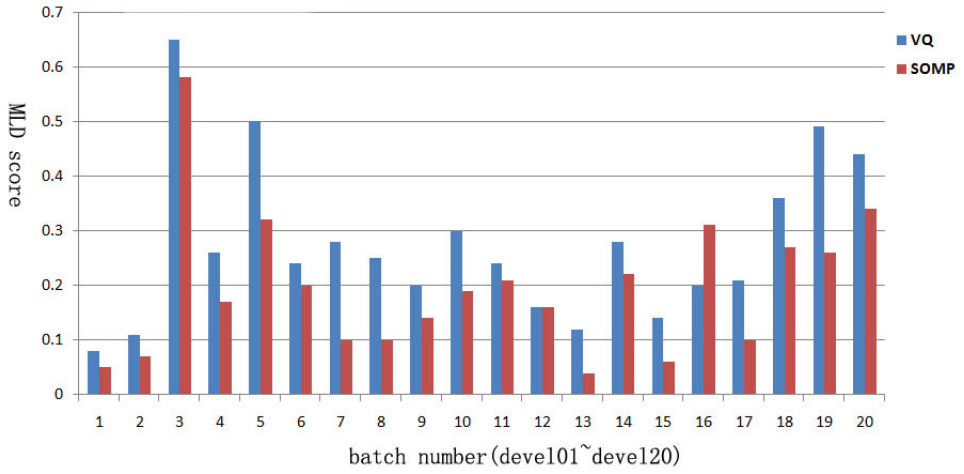


Figure 16: Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$, $k = 10$ and $\gamma = 0.3$ ($devel01 \sim devel20$). The results with different coding methods (VQ, SOMP).

4.4.2 COMPARISON BETWEEN VQ AND SOMP

We then evaluate different coding methods (VQ, SOMP) on development ($devel01 \sim devel20$) batches. Figure 16 shows the results. The minimum MLD by SOMP is 0.004 (see $devel13$), while 0.008 (see $devel01$) for VQ. And most of the performances by SOMP are much better than VQ. Later, we test 3D MoSIFT and 3D EMoSIFT features on $final21 \sim final40$ batches. MLD scores are given in Table 6. It can be seen that in most cases, SOMP leads the performance whenever 3D MoSIFT or 3D EMoSIFT is used. We also provide the results by 3D EMoSIFT for every batch in Figure 17 which shows that SOMP is better than VQ in most cases. In a word, compared with VQ, SOMP not only has lower reconstruction errors (see Figure 9) but also achieves better performance. We note that 3D EMoSIFT does not work well on $devel03$ batch as shown in Figure 16. That is because there are static gestures (postures) on $devel03$ batch, while 3D EMoSIFT can only capture distinctive features when the gestures are in motion.

Methods \ γ	0.1	0.2	0.3	0.4	0.5
3D MoSIFT_VQ	0.19135	0.16694	0.16195	0.14476	0.14642
3D MoSIFT_SOMP	**0.18303**	**0.16251**	**0.15918**	0.15086	**0.14088**
3D EMoSIFT_VQ	0.16528	0.15419	0.14753	0.13977	0.13311
3D EMoSIFT_SOMP	0.17415	**0.14753**	**0.14032**	**0.13478**	**0.13145**

Table 6: Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$, $k = 10$, and γ varies from 0.1 to 0.5 ($final21 \sim final40$). MLD scores are calculated by different coding methods.

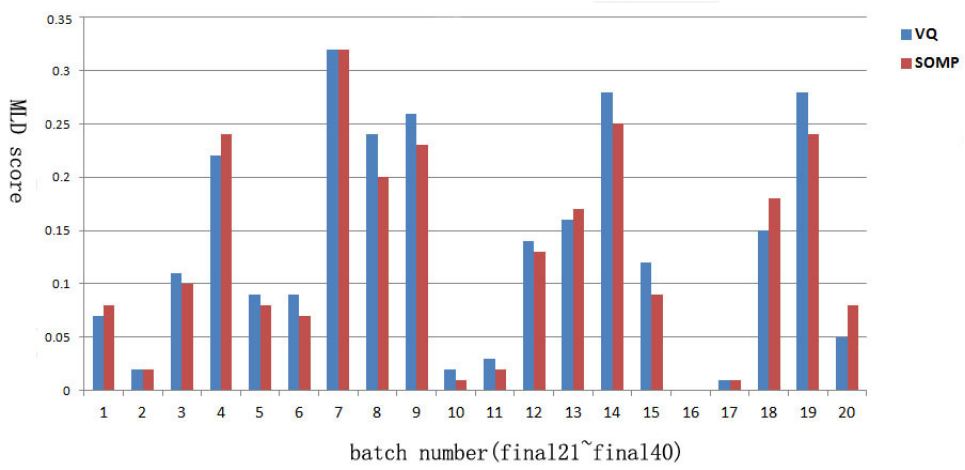


Figure 17: Parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$, $k = 10$ and $\gamma = 0.3$ ($final21 \sim final40$). The results with different coding methods (VQ, SOMP).

4.4.3 COMPARISON BETWEEN GRADIENT-BASED AND MOTION-BASED FEATURES

We know that 3D EMoSIFT feature includes two basic components, namely, gradient-based features and motion-based features. And each component is of size 384 dimensions. In this section, we separately evaluate these two components and determinate which component is more essential to gesture recognition. The results evaluated on development batches are separately shown in Figure 18 where the integrated feature consists of the gradient-based and motion-based features. The average MLD scores are 0.1945 for the integrated feature, 0.216 for the gradient-based features, and 0.313 for the motion-based features. It can be seen that the performance of the gradient-based features, which are comparative to the results of the integrated feature, are much better than the performance of the motion-based features. In addition, our method outperforms two published papers on $devel01 \sim devel20$ batches, that is say, our method: 0.1945, Lui (2012): 0.2873, Malgireddy et al. (2012): 0.2409.

As mentioned in Section 3.1, 3D EMoSIFT is constructed in two stages (interest point detection and descriptor calculation). So whenever the gradient-based or motion-based features are calculated, we should first detect the interest points. We randomly select a sample from Chalearn gesture database and test the average time with c++ programs and OpenCV library (Bradski, 2000) on a standard personal computer (CPU: 3.3GHz, RAM: 8GB). Table 7 shows that the main processing time occurs in the stage of interest point detection. The remaining parts for calculating the gradient-base and motion-based descriptor is small compared with the time for interest point detection. In our future work, we will focus on how to efficiently detect interest points.

4.4.4 COMPARISON WITH OTHER METHODS

Here, we compare the proposed approach with some popular sequence matching methods such as HMM, DTW, CRF, HCRF and LDCRF, and also give the final results of top contestants. The results are reported in Table 8 where the principal motion method (Escalante and Guyon, 2012) is the baseline method and DTW is an optional method on Chalearn gesture challenge (round 2).

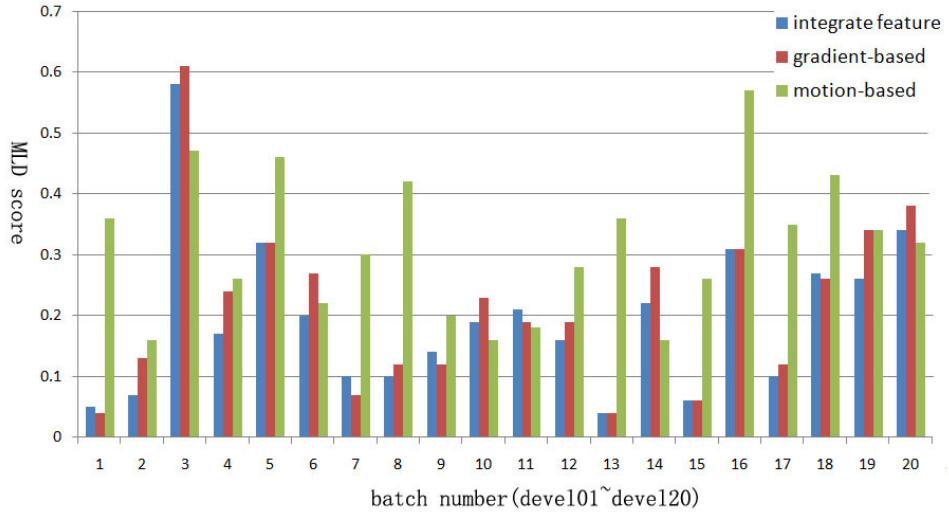


Figure 18: Results for parameters: $\beta_1 = \beta_2 = 0.005$, $\sigma = 1.6$, $s = 3$, $k = 10$, and $\gamma = 0.3$ (*devel*01 $\sim$ *devel*20).

interest point detection average time (ms/f)	gradient-based descriptor average time (ms/f)	motion-based descriptor average time (ms/f)
887	2.1	1.4

Table 7: The average computation time for different parts in 3D EMoSIFT feature.

method	validation set(01 $\sim$ 20)	final set(21 $\sim$ 40)	team name
motion signature analysis	0.0995	0.0710	Alfnie
HMM+HOGHOF	0.2084	0.1098	Turtle Tamers
BoF+3D MoSIFT	0.1824	0.1448	Joewan
principle motion	0.3418	0.3172	–
DTW	0.4616	0.3899	–
CRF	0.6365	0.528	–
HCRF	0.64	0.6	–
LDCRF	0.608	0.5145	–
our method	**0.1595**	**0.1259**	–

Table 8: Results of different methods on Chalearn gesture data set.

The top ranking results in the competition are from three teams (Alfine, Turtle Tamers and Joewan), which are provided in the technical report (Guyon et al., 2013). We use the code provided by Morency et al. (2007) to train the CRF-based classifiers, because this code was well developed and can be easily used. Every frame is represented by a vector of motion feature mentioned in Section 3.4. Those motion features extracted from training videos are used to train CRF-based

models. For the CRF model, every class has a corresponding label (gesture label). CRF predicts a label for each frame in a video. During evaluation, the video label is predicted based on the most frequently occurring label per frame (Morency et al., 2007). For the HCRF (or LDCRF) model, we train a single HCRF (or LDCRF) model with different number of hidden states (from 2 to 6 states) and select the lowest MLD scores as the final results which are shown in Table 8. We can see that the proposed method is competitive to the state-of-the-art methods. Besides, the CRF-based methods get poor performances. That is because the simple motion features may be indistinguishable to represent the gesture pattern.

5. Conclusion

In this paper, we propose a unified framework based on bag of features for one-shot learning gesture recognition. The proposed method gives superior recognition performance than many existing approaches. A new feature, named 3D EMoSIFT, fuses RGB-D data to detect interest points and constructs 3D gradient and motion space to calculate SIFT descriptors. Compared with existing features such as Cuboid (Dollár et al., 2005), Harri3D (Laptev, 2005), MoSIFT (Chen and Hauptmann, 2009) and 3D MoSIFT (Ming et al., 2012), it gets competitive performance. Additionally, 3D EMoSIFT features are scale and rotation invariant and can capture more compact and richer video representations even though there is only one training sample for each gesture class. This paper also introduces SOMP to replace VQ in the descriptor coding stage. Then each feature can be represented by some linear combination of a small number of visual codewords. Compared with VQ, SOMP leads to a much lower reconstruction error and achieves better performance.

Although the proposed method has achieved promising results, there are several avenues which can be explored. At first, most of the existing local spatio-temporal features are extracted from a static background or a simple dynamic background. In our feature research, we will focus on extending 3D EMoSIFT to extract features from complex background, especially for one-shot learning gesture recognition. Next, to speed up processing time, we can achieve fast feature extraction on a Graphics Processing Unit (GPU) (Chen et al., 2003). Also, we will explore the techniques required to optimize the parameters, such as the codebook size and sparsity.

Acknowledgments

We appreciate ChaLearn providing the gesture database (`http://chalearn.org`) whose directors are gratefully acknowledged. We would like to thank Isabelle Guyon, ChaLearn, Berkeley, California, who gives us insightful comments and suggestions to improve our manuscripts. And we are grateful to editors and anonymous reviewers whose instructive suggestions have improved the quality of this paper. Besides, thanks to acknowledge support for this project from National Natural Science Foundation (60973060, 61003114, 61172128), National 973 plans project (2012CB316304), the fundamental research funds for the central universities (2011JBM020, 2011JBM022) and the program for Innovative Research Team in University of Ministry of Education of China (IRT 201206).

References

G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999, 1997.

C.C. Chang and C.J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011.

F.S. Chen, C.M. Fu, and C.L. Huang. Hand gesture recognition using a real-time tracking method and hidden markov models. *Image and Vision Computing*, 21:745–758, 2003.

M. Chen and A. Hauptmann. Mosift: Recognizing human actions in surveillance videos. *Technical Report*, 2009.

H. Cooper, E.J. Ong, N. Pugeault, and R. Bowden. Sign language recognition using sub-units. *Journal of Machine Learning Research*, 13:2205–2231, 2012.

A. Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, IEEE ICCV Workshop on*, pages 82–89, 2001.

N.H. Dardas and N.D. Georganas. Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. *IEEE Transactions on Instrumentation and Measurement*, 60(11):3592–3607, 2011.

P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *Proceedings of IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005.

H.J. Escalante and I. Guyon. Principal motion: Pca-based reconstruction of motion histograms. *Technical Memorandum*, 2012.

L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 524–531, 2005.

F. Flórez, J.M. García, J. García, and A. Hernández. Hand gesture recognition following the dynamics of a topology-preserving network. In *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, pages 318–323, 2002.

P.-E. Forssen and D.G. Lowe. Shape descriptors for maximally stable extremal regions. In *Computer Vision, IEEE 11th International Conference on*, pages 1–8, 2007.

W.T. Freeman and M. Roth. Orientation histograms for hand gesture recognition. In *Proceedings of IEEE International Workshop on Automatic Face and Gesture Recognition*, volume 12, pages 296–301, 1995.

W. Gao, G. Fang, D. Zhao, and Y. Chen. A chinese sign language recognition system based on sofm/srn/hmm. *Pattern Recognition*, 37(12):2389–2402, 2004.

T. Guha and R.K. Ward. Learning sparse representations for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1576–1588, 2012.

S. Guo, Z. Wang, and Q. Ruan. Enhancing sparsity via ℓ_p (0<p<1) minimization for robust face recognition. *Neurocomputing*, 99:592–602, 2013.

I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hamner, and H.J. Escalante. Chalearn gesture challenge: Design and first results. In *Computer Vision and Pattern Recognition Workshops, IEEE Conference on*, pages 1–6, 2012.

I. Guyon, V. Athitsos, P. Jangyodsuk, H.J. Escalante, and B. Hamner. Results and analysis of the chalearn gesture challenge 2012. *Technical Report*, 2013.

C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of Alvey Vision Conference*, volume 15, page 50, 1988.

A. Hernández-Vela, M. A. Bautista, X. Perez-Sala, V. Ponce, X. Baró, O. Pujol, C. Angulo, and S. Escalera. Bovdw: Bag-of-visual-and-depth-words for gesture recognition. *Pattern Recognition (ICPR), 21st International Conference on*, 2012.

D. Kim, J. Song, and D. Kim. Simultaneous gesture segmentation and recognition based on forward spotting accumulative hmms. *Pattern Recognition*, 40(11):3012–3026, 2007.

I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2):107–123, 2005.

J.F. Lichtenauer, E.A. Hendriks, and M. J T Reinders. Sign language recognition by combining statistical dtw and independent classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(11):2040–2046, 2008.

Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *Communications, IEEE Transactions on*, 28(1):84–95, 1980.

D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

B.D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 1981.

Yui Man Lui. Human gesture recognition on product manifolds. *Journal of Machine Learning Research*, 13:3297–3321, 2012.

M.R. Malgireddy, I. Inwogu, and V. Govindaraju. A temporal bayesian model for classifying, detecting and localizing activities in video sequences. In *Computer Vision and Pattern Recognition Workshops, IEEE Conference on*, pages 43–48, 2012.

A. Malima, E. Ozgur, and M. Çetin. A fast algorithm for vision-based hand gesture recognition for robot control. In *Proceedings of IEEE Signal Processing and Communications Applications*, pages 1–4, 2006.

Y. Ming, Q. Ruan, and A.G. Hauptmann. Activity recognition from rgb-d camera with 3d local spatio-temporal features. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 344–349, 2012.

L.P. Morency, A. Quattoni, and T. Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

B.A. Olshausen, D.J. Field, et al. Sparse coding with an overcomplete basis set: A strategy employed by vi? *Vision Research*, 37(23):3311–3326, 1997.

V.I. Pavlovic, R. Sharma, and T.S. Huang. Visual interpretation of hand gestures for human-computer interaction: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:677–695, 1997.

A. Rakotomamonjy. Surveying and comparing simultaneous sparse approximation (or group-lasso) algorithms. *Signal Processing*, 91(7):1505–1526, 2011.

S. Reifinger, F. Wallhoff, M. Ablassmeier, T. Poitschke, and G. Rigoll. Static and dynamic hand-gesture recognition for augmented reality applications. In *Proceedings of the 12th International Conference on Human-computer Interaction: Intelligent Multimodal Interaction Environments*, pages 728–737, 2007.

Y. Ruiduo, S. Sarkar, and B. Loeding. Enhanced level building algorithm for the movement epenthesis problem in sign language recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

C. Shan, T. Tan, and Y. Wei. Real-time hand tracking using a mean shift embedded particle filter. *Pattern Recognition*, 40(7):1958–1970, 2007.

X. Shen, G. Hua, L. Williams, and Y. Wu. Dynamic hand gesture recognition: An exemplar-based approach from motion divergence fields. *Image and Vision Computing*, 30(3):227–235, 2012.

C. Sminchisescu, A. Kanaujia, Zhiguo Li, and D. Metaxas. Conditional models for contextual human motion recognition. In *Computer Vision, Tenth IEEE International Conference on*, volume 2, pages 1808–1815, 2005.

H.I. Suk, B.K. Sin, and S.W. Lee. Hand gesture recognition based on dynamic bayesian network framework. *Pattern Recognition*, 43(9):3059–3072, 2010.

J. Weaver T. Starner and A. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1371–1375, 1998.

J.A. Tropp, A.C. Gilbert, and M.J. Strauss. Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *Signal Processing*, 86(3):572–588, 2006.

A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, 13(2):260–269, 1967.

C. P. Vogler. *American Sign Language Recognition: Reducing the Complexity of the Task with Phoneme-based Modeling and Parallel Hidden Markov Models*. PhD thesis, Doctoral dissertation, University of Pennsylvania, 2003.

J. Wan, Q. Ruan, G. An, and W. Li. Gesture recognition based on hidden markov model from sparse representative observations. In *Signal Processing (ICSP), IEEE 10th International Conference on*, pages 1180–1183, 2012.

H. Wang, M.M. Ullah, A. Klaser, I. Laptev, C. Schmid, et al. Evaluation of local spatio-temporal features for action recognition. In *Proceedings of British Machine Vision Conference*, 2009.

J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3360–3367, 2010.

S.B. Wang, A. Quattoni, L.P. Morency, D. Demirdjian, and T. Darrell. Hidden conditional random fields for gesture recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1521–1527, 2006.

J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:210–227, 2009.

J. Yamato, Jun Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801, 2009.

M.H. Yang, N. Ahuja, and M. Tabb. Extraction of 2d motion trajectories and its application to hand gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24: 1061–1074, 2002.

D. Youtian, C. Feng, X. Wenli, and Li. Yongbin. Recognizing interaction activities using dynamic bayesian network. In *Pattern Recognition, 18th International Conference on*, volume 1, pages 618–621, 2006.

Y. Zhu, G. Xu, and D.J. Kriegman. A real-time approach to the spotting, representation, and recognition of hand gestures for human–computer interaction. *Computer Vision and Image Understanding*, 85(3):189–208, 2002.

Journal of Machine Learning Research 14 (2013) 2583-2615 Submitted 12/12; Revised 5/13; Published 9/13

Efficient Active Learning of Halfspaces: An Aggressive Approach

Alon Gonen ALONGNN@CS.HUJI.AC.IL
The Rachel and Selim Benin School of Computer Science and Engineering
The Hebrew University
Givat Ram, Jerusalem 91904, Israel

Sivan Sabato SIVAN.SABATO@MICROSOFT.COM
Microsoft Research New England
1 Memorial Drive
Cambridge, MA, 02142

Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
The Rachel and Selim Benin School of Computer Science and Engineering
The Hebrew University
Givat Ram, Jerusalem 91904, Israel

Editor: John Shawe-Taylor

Abstract

We study pool-based active learning of half-spaces. We revisit the aggressive approach for active learning in the realizable case, and show that it can be made efficient and practical, while also having theoretical guarantees under reasonable assumptions. We further show, both theoretically and experimentally, that it can be preferable to mellow approaches. Our efficient aggressive active learner of half-spaces has formal approximation guarantees that hold when the pool is separable with a margin. While our analysis is focused on the realizable setting, we show that a simple heuristic allows using the same algorithm successfully for pools with low error as well. We further compare the aggressive approach to the mellow approach, and prove that there are cases in which the aggressive approach results in significantly better label complexity compared to the mellow approach. We demonstrate experimentally that substantial improvements in label complexity can be achieved using the aggressive approach, for both realizable and low-error settings.

Keywords: active learning, linear classifiers, margin, adaptive sub-modularity

1. Introduction

We consider pool-based active learning (McCallum and Nigam, 1998), in which a learner receives a pool of unlabeled examples, and can iteratively query a teacher for the labels of examples from the pool. The goal of the learner is to return a low-error prediction rule for the labels of the examples, using a small number of queries. The number of queries used by the learner is termed its *label complexity*. This setting is most useful when unlabeled data is abundant but labeling is expensive, a common case in many data-laden applications. A pool-based algorithm can be used to learn a classifier in the standard PAC model, while querying fewer labels. This can be done by first drawing a random unlabeled sample to be used as the pool, then using pool-based active learning to identify its labels with few queries, and then using the resulting labeled sample as input to a regular "passive" PAC-learner.

Most active learning approaches can be loosely described as more 'aggressive' or more 'mellow'. A more aggressive approach is one in which only highly informative queries are requested (where the meaning of 'highly informative' depends on the particular algorithm) (Tong and Koller, 2002; Balcan et al., 2007; Dasgupta et al., 2005), while the mellow approach, first proposed in the CAL algorithm (Cohn et al., 1994), is one in which the learner essentially queries all the labels it has not inferred yet.

In recent years a significant advancement has been made for active learning in the PAC model. In particular, it has been shown that when the data is realizable (relative to some assumed hypothesis class), the mellow

approach can guarantee an exponential improvement in label complexity, compared to passive learning (Balcan et al., 2006a). This exponential improvement depends on the properties of the distribution, as quantified by the *Disagreement Coefficient* proposed in Hanneke (2007). Specifically, when learning half-spaces in Euclidean space, the disagreement coefficient implies a low label complexity when the data distribution is uniform or close to uniform. Guarantees have also been shown for the case where the data distribution is a finite mixture of Gaussians (El-Yaniv and Wiener, 2012).

An advantage of the mellow approach is its ability to obtain label complexity improvements in the agnostic setting, which allows an arbitrary and large labeling error (Balcan et al., 2006a; Dasgupta et al., 2007). Nonetheless, in the realizable case the mellow approach is not always optimal, even for the uniform distribution (Balcan et al., 2007). In this work we revisit the aggressive approach for the realizable case, and in particular for active learning of half-spaces in Euclidean space. We show that it can be made efficient and practical, while also having theoretical guarantees under reasonable assumptions. We further show, both theoretically and experimentally, that it can sometimes be preferable to mellow approaches.

In the first part of this work we construct an efficient aggressive active learner for half-spaces in Euclidean space, which is approximately optimal, that is, achieves near-optimal label complexity, if the pool is separable with a margin. While our analysis is focused on the realizable setting, we show that a simple heuristic allows using the same algorithm successfully for pools with low error as well. Our algorithm for halfspaces is based on a greedy query selection approach as proposed in Tong and Koller (2002) and Dasgupta (2005). We obtain improved target-dependent approximation guarantees for greedy selection in a general active learning setting. These guarantees allow us to prove meaningful approximation guarantees for halfspaces based on a margin assumption.

In the second part of this work we compare the greedy approach to the mellow approach. We prove that there are cases in which this highly aggressive greedy approach results in significantly better label complexity compared to the mellow approach. We further demonstrate experimentally that substantial improvements in label complexity can be achieved compared to mellow approaches, for both realizable and low-error settings.

The first greedy query selection algorithm for learning halfspaces in Euclidean space was proposed by Tong and Koller (2002). The greedy algorithm is based on the notion of a *version space*: the set of all hypotheses in the hypothesis class that are consistent with the labels currently known to the learner. In the case of halfspaces, each version space is a convex body in Euclidean space. Each possible query thus splits the current version space into two parts: the version space that would result if the query received a positive label, and the one resulting from a negative label. Tong and Koller proposed to query the example from the pool that splits the version space as evenly as possible. To implement this policy, one would need to calculate the volume of a convex body in Euclidean space, a problem which is known to be computationally intractable (Brightwell and Winkler, 1991). Tong and Koller thus implemented several heuristics that attempt to follow their proposed selection principle using an efficient algorithm. For instance, they suggest to choose the example which is closest to the max-margin solution of the data labeled so far. However, none of their heuristics provably follow this greedy selection policy.

The label complexity of greedy pool-based active learning algorithms can be analyzed by comparing it to the best possible label complexity of any pool-based active learner on the same pool. The *worst-case label complexity* of an active learner is the maximal number of queries it would make on the given pool, where the maximum is over all the possible classification rules that can be consistent with the pool according to the given hypothesis class. The *average-case label complexity* of an active learner is the average number of queries it would make on the given pool, where the average is taken with respect to some fixed probability distribution P over the possible classifiers in the hypothesis class. For each of these definitions, the optimal label complexity is the lowest label complexity that can be achieved by an active learner on the given pool. Since implementing the optimal label complexity is usually computationally intractable, an alternative is to implement an efficient algorithm, and to guarantee a bounded factor of approximation on its label complexity, compared to the optimal label complexity.

Dasgupta (2005) showed that if a greedy algorithm splits the probability mass of the version space as evenly as possible, as defined by the fixed probability distribution P over the hypothesis class, then the approximation factor for its average label complexity, with respect to the same distribution, is bounded by

$O(\log(1/p_{\min}))$, where $p_{\min}$ is the minimal probability of any possible labeling of the pool, if the classifier is drawn according to the fixed distribution. Golovin and Krause (2010) extended Dasgupta's result and showed that a similar bound holds for an approximate greedy rule. They also showed that the approximation factor for the worst-case label complexity of an approximate greedy rule is also bounded by $O(\log(1/p_{\min}))$, thus extending a result of Arkin et al. (1993). Note that in the worst-case analysis, the fixed distribution is only an analysis tool, and does not represent any assumption on the true probability of the possible labelings.

Returning to greedy selection of halfspaces in Euclidean space, we can see that the fixed distribution over hypotheses that matches the volume-splitting strategy is the distribution that draws a halfspace uniformly from the unit ball.[1] The analysis presented above thus can result in poor approximation factors, since if there are instances in the pool that are very close to each other, then $p_{\min}$ might be very small.

We first show that mild conditions suffice to guarantee that $p_{\min}$ is bounded from below. By proving a variant of a result due to Muroga et al. (1961), we show that if the examples in the pool are stored using number of a finite accuracy $1/c$, then $p_{\min} \geq (c/d)^{d^2}$, where d is the dimensionality of the space. It follows that the approximation factor for the worst-case label complexity of our algorithm is at most $O(d^2 \log(d/c))$.

While this result provides us with a uniform lower bound on $p_{\min}$, in many real-world situations the probability of the target hypothesis (i.e., one that is consistent with the true labeling) could be much larger than $p_{\min}$. A noteworthy example is when the target hypothesis separates the pool with a margin of γ. In this case, it can be shown that the probability of the target hypothesis is at least γ^d, which can be significantly larger than $p_{\min}$. An immediate question is therefore: can we obtain a *target-dependent* label complexity approximation factor that would depend on the probability of the target hypothesis, $P(h)$, instead of the minimal probability of any labeling?

We prove that such a target dependent bound *does not* hold for a general approximate-greedy algorithm. To overcome this, we introduce an algorithmic change to the approximate greedy policy, which allows us to obtain a label complexity approximation factor of $\log(1/P(h))$. This can be achieved by running the approximate-greedy procedure, but stopping the procedure early, before reaching a pure version space that exactly matches the labeling of the pool. Then, an approximate majority vote over the version space, that is, a random rule which approximates the majority vote with high probability, can be used to determine the labels of the pool. This result is general and holds for any hypothesis class and distribution. For halfspaces, it implies an approximation-factor guarantee of $O(d \log(1/\gamma))$.

We use this result to provide an efficient approximately-optimal active learner for half-spaces, called *ALuMA*, which relies on randomized approximation of the volume of the version space (Kannan et al., 1997). This allows us to prove a margin-dependent approximation factor guarantee for ALuMA. We further show an additional, more practical implementation of the algorithm, which has similar guarantees under mild conditions which often hold in practice. The assumption of separation with a margin can be relaxed if a lower bound on the total hinge-loss of the best separator for the pool can be assumed. We show that under such an assumption a simple transformation on the data allows running ALuMA as if the data was separable with a margin. This results in approximately optimal label complexity with respect to the new representation.

We also derive lower bounds, showing that the dependence of our label-complexity guarantee on the accuracy c, or the margin parameter γ, is indeed necessary and is not an artifact of our analysis. We do not know if the dependence of our bounds on d is tight. It should be noted that some of the most popular learning algorithms (e.g., SVM, Perceptron, and AdaBoost) rely on a large-margin assumption to derive dimension-independent sample complexity guarantees. In contrast, here we use the margin for computational reasons. Our approximation guarantee depends logarithmically on the margin parameter, while the sample complexities of SVM, Perceptron, and AdaBoost depend polynomially on the margin. Hence, we require a much smaller margin than these algorithms do. In a related work, Balcan et al. (2007) proposed an active learning algorithm with dimension-independent guarantees under a margin assumption. These guarantees hold for a restricted class of data distributions.

In the second part of this work, we compare the greedy approach to the mellow approach of CAL in the realizable case, both theoretically and experimentally. Our theoretical results show the following:

1. We discuss the challenges presented by other natural choices of a distribution in Section 2.

1. In the simple learning setting of thresholds on the line, our margin-based approach is preferable to the mellow approach when the true margin of the target hypothesis is large.

2. There exists a distribution in Euclidean space such that the mellow approach cannot achieve a significant improvement in label complexity over passive learning for halfspaces, while the greedy approach achieves such an improvement using more unlabeled examples.

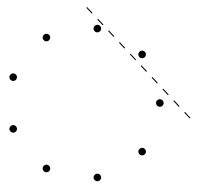
3. There exists a pool in Euclidean space such that the mellow approach requires exponentially more labels than the greedy approach.

We further compare the two approaches experimentally, both on separable data and on data with small error. The empirical evaluation indicates that our algorithm, which can be implemented in practice, achieves state-of-the-art results. It further suggests that aggressive approaches can be significantly better than mellow approaches in some practical settings.

2. On the Challenges in Active Learning for Halfspaces

The approach we employ for active learning does not provide absolute guarantees for the label complexity of learning, but a relative guarantee instead, in comparison with the optimal label complexity. One might hope that an absolute guarantee could be achieved using a different algorithm, for instance in the case of half-spaces. However, the following example from Dasgupta (2005) indicates that no meaningful guarantee can be provided that holds for all possible pools.

Example 1 *Consider a distribution in $\mathbb{R}^d$ for any $d \geq 3$. Suppose that the support of the distribution is a set of evenly-distributed points on a two-dimensional sphere that does not circumscribe the origin, as illustrated in the following figure. As can be seen, each point can be separated from the rest of the points with a halfspace.*



In this example, to distinguish between the case in which all points have a negative label and the case in which one of the points has a positive label while the rest have a negative label, any active learning algorithm will have to query every point at least once. It follows that for any $\varepsilon > 0$, if the number of points is $1/\varepsilon$, then the label complexity to achieve an error of at most ε is $1/\varepsilon$. On the other hand, the sample complexity of passive learning in this case is order of $\frac{1}{\varepsilon}\log\frac{1}{\varepsilon}$, hence no active learner can be significantly better than a passive learner on this distribution.

Since we provide margin-dependent guarantees, one may wonder if a margin assumption alone can guarantee that few queries suffice to learn the half-space. This is not the case, as evident by the following variation of Example 1.

Example 2 *Let $\gamma \in (0, \frac{1}{2})$ be a margin parameter. Consider a pool of m points in $\mathbb{R}^d$, such that all the points are on the unit sphere, and for each pair of points x_1 and x_2, $\langle x_1, x_2\rangle \leq 1 - 2\gamma$. It was shown in Shannon (1959) that for any $m \leq O(1/\gamma^d)$, there exists a set of points that satisfy the conditions above. For any point x in such a pool, there exists a (biased) halfspace that separates x from the rest of the points with a margin of γ. This can be seen by letting $w = x$ and $b = 1 - \gamma$. Then $\langle w, x\rangle - b = \gamma$ while for any $z \neq x$ in the set, $\langle w, z\rangle - b = \langle x, z\rangle - 1 + \gamma \leq -\gamma$. By adding a single dimension, this example can be transformed to one with homogeneous (unbiased) halfspaces. Each point in this pool can be separated from the rest of the points by a halfspace. Thus, if the correct labeling is all-positive, then all m examples need to be queried to label the pool correctly.*

These examples show that there are "difficult" pools, where no active learner can do well. The advantage of the greedy approach is that the optimal label complexity is used as a natural measure of the difficulty of the pool.

At first glance it might seem that there are simpler ways to implement an efficient greedy strategy for halfspaces, by using a different distribution over the hypotheses. For instance, if there are m examples in d dimensions, Sauer's lemma states that the effective size of the hypothesis class of halfspaces will be at most m^d. One can thus use the uniform distribution over this finite class, and greedily reduce the number of possible hypotheses in the version space, obtaining a $d\log(m)$ factor relative to the optimal label complexity. However, a direct implementation of this method will be exponential in d, and it is not clear whether this approach has a polynomial implementation.

Another approach is to discretize the version space, by considering only halfspaces that can be represented as vectors on a d-dimensional grid $\{-1,-1+c,\ldots,1-c,1\}^d$. This results in a finite hypothesis class of size $(2/c+1)^d$, and we get an approximation factor of $O(d\log(1/c))$ for the greedy algorithm, compared to an optimal algorithm on the same finite class. However, it is unknown whether a greedy algorithm for reducing the number of such vectors in a version space can be implemented efficiently, since even determining whether a single grid point exists in a given version space is NP-hard (see, e.g., Matoušek, 2002, Section 2.2). In particular, the volume of the version space cannot be used to estimate this quantity, since the volume of a body and the number of grid points in this body are not correlated. For example, consider a line in $\mathbb{R}^2$, whose volume is 0. It can contain zero grid points or many grid points, depending on its alignment with respect to the grid. Therefore, the discretization approach is not straightforward as one might first assume. In fact, if this approach is at all computationally feasible, it would probably require the use of some approximation scheme, similarly to the volume-estimation approach that we describe below.

Yet another possible direction for pool-based active learning is to greedily select a query whose answer would determine the labels of the largest amount of pool examples. The main challenge in this direction is how to analyze the label complexity of such an algorithm: it is unclear whether competitiveness with the optimal label complexity can be guaranteed in this case. Investigating this idea, both theoretically and experimentally, is an important topic for future work. Note that the CAL algorithm (Cohn et al., 1994), which we discuss in Section 6, can be seen as implementing a mellow version of this approach, since it decreases the so-called "disagreement region" in each iteration.

3. Definitions and Preliminaries

In pool-based active learning, the learner receives as input a set of instances, denoted $X=\{x_1,\ldots,x_m\}$. Each instance x_i is associated with a label $L(i)\in\{\pm 1\}$, which is initially unknown to the learner. The learner has access to a teacher, represented by the oracle $L:[m]\to\{-1,1\}$. An active learning algorithm $\mathcal{A}$ obtains (X,L,T) as input, where T is an integer which represents the label budget of $\mathcal{A}$. The goal of the learner is to find the values $L(1),\ldots,L(m)$ using as few calls to L as possible. We assume that L is determined by a function h taken from a predefined hypothesis class $\mathcal{H}$. Formally, for an oracle L and a hypothesis $h\in\mathcal{H}$, we write $L\Leftarrow h$ to state that for all i, $L(i)=h(x_i)$.

Given $S\subseteq X$ and $h\in\mathcal{H}$, we denote the partial realization of h on S by

$$h|_S=\{(x,h(x)):x\in S\}\ .$$

We denote by $V(h|_S)$ the version space consisting of the hypotheses which are consistent with $h|_S$. Formally,

$$V(h|_S)=\{h'\in\mathcal{H}:\forall x\in S,\ \ h'(x)=h(x)\ \}.$$

Given X and $\mathcal{H}$, we define, for each $h\in\mathcal{H}$, the equivalence class of h over $\mathcal{H}$, $[h]=\{h'\in\mathcal{H}\mid\forall x\in X, h(x)=h'(x)\}$. We consider a probability distribution P over $\mathcal{H}$ such that $P([h])$ is defined for all $h\in\mathcal{H}$. For brevity, we denote $P(h)=P([h])$. Similarly, for a set $V\subseteq\mathcal{H}$, $P(V)=P(\cup_{h\in V}[h])$. Let $p_{\min}=\min_{h\in\mathcal{H}}P(h)$.

We specifically consider the hypothesis class of homogeneous halfspaces in $\mathbb{R}^d$. In this case, $X\subseteq\mathbb{R}^d$. The hypothesis class $\mathcal{H}$ is defined by $\mathcal{W}=\{x\mapsto\operatorname{sgn}(\langle w,x\rangle)\mid w\in\mathbb{R}^d\}$, where $\langle w,x\rangle$ is the inner product between the vectors w and x.

For a given active learning algorithm $\mathcal{A}$, we denote by $N(\mathcal{A},h)$ the number of calls to L that $\mathcal{A}$ makes before outputting $(L(x_1),\ldots,L(x_m))$, under the assumption that $L \Leftarrow h$. The worst-case label complexity of $\mathcal{A}$ is defined to be

$$c_{\mathrm{wc}}(\mathcal{A}) \stackrel{\mathrm{def}}{=} \max_{h\in\mathcal{H}} N(\mathcal{A},h).$$

We denote the optimal worst-case label complexity for the given pool by $\mathrm{OPT}_{\max}$. Formally, we define $\mathrm{OPT}_{\max} = \min_{\mathcal{A}} c_{\mathrm{wc}}(\mathcal{A})$, where the minimum is taken over all possible active learners for the given pool.

Given a probability distribution P over $\mathcal{H}$, the average-case label complexity of $\mathcal{A}$ is defined to be

$$c_{\mathrm{avg}}(\mathcal{A}) \stackrel{\mathrm{def}}{=} \mathbb{E}_{h\sim P} N(\mathcal{A},h).$$

The optimal average label complexity for the given pool X and probability distribution P is defined as $\mathrm{OPT}_{\mathrm{avg}} = \min_{\mathcal{A}} c_{\mathrm{avg}}(\mathcal{A})$.

For a given active learner, we denote by $V_t \subseteq \mathcal{H}$ the version space of an active learner after t queries. Formally, suppose that the active learning queried instances $i_1,\ldots,i_t$ in the first t iterations. Then

$$V_t = \{h \in \mathcal{H} \mid \forall j \in [t], h(x_{i_j}) = L(i_j)\}.$$

For a given pool example $x \in X$, denote by $V_{t,x}^j$ the version spaces that would result if the algorithm now queried x and received label j. Formally,

$$V_{t,x}^j = V_t \cap \{h \in \mathcal{H} \mid h(x) = j\}.$$

A greedy algorithm (with respect to a probability distribution P) is an algorithm $\mathcal{A}$ that at each iteration $t = 1,\ldots,T$, the pool example x that $\mathcal{A}$ decides to query is one that splits the version space as evenly as possible. Formally, at every iteration t $\mathcal{A}$ queries some example in $\operatorname{argmin}_{x\in X} \max_{j\in\{\pm 1\}} P(V_{t,x}^j)$. Equivalently, a greedy algorithm is an algorithm $\mathcal{A}$ that at every iteration t queries an example in

$$\operatorname*{argmax}_{x\in X} P(V_{t,x}^{-1}) \cdot P(V_{t,x}^{+1}).$$

To see the equivalence, note that $P(V_{t,x}^{-1}) = P(V_t) - P(V_{t,x}^{+1})$. Therefore,

$$P(V_{t,x}^{-1}) \cdot P(V_{t,x}^{+1}) = (P(V_t) - P(V_{t,x}^{+1}))P(V_{t,x}^{+1}) = (P(V_t)/2)^2 - (P(V_t)/2 - P(V_{t,x}^{+1}))^2.$$

It follows that the expression is monotonic decreasing in $|P(V_t)/2 - P(V_{t,x}^{+1})|$.

This equivalent formulation motivates the following definition of an approximately greedy algorithm, following Golovin and Krause (2010).

Definition 3 *An algorithm $\mathcal{A}$ is called* α-approximately greedy *with respect to P, for $\alpha \geq 1$, if at each iteration $t = 1,\ldots,T$, the pool example x that $\mathcal{A}$ decides to query satisfies*

$$P(V_{t,x}^1)P(V_{t,x}^{-1}) \geq \frac{1}{\alpha} \max_{\tilde{x}\in X} P(V_{t,\tilde{x}}^1)P(V_{t,\tilde{x}}^{-1}),$$

and the output of the algorithm is $(h(x_1),\ldots,h(x_m))$ for some $h \in V_T$.

It is easy to see that by this definition, an algorithm is exactly greedy if it is approximately greedy with $\alpha = 1$.

By Dasgupta (2005) we have the following guarantee: For any exactly greedy algorithm $\mathcal{A}$ with respect to distribution P,

$$c_{\mathrm{avg}}(\mathcal{A}) = O(\log(1/p_{\min}) \cdot \mathrm{OPT}_{\mathrm{avg}}).$$

Golovin and Krause (2010) show that for an α approximately greedy algorithm,

$$c_{\mathrm{avg}}(\mathcal{A}) = O(\alpha \cdot \log(1/p_{\min}) \cdot \mathrm{OPT}_{\mathrm{avg}}).$$

In addition, they show a similar bound for the worst-case label complexity. Formally,

$$c_{\mathrm{wc}}(\mathcal{A}) = O(\alpha \cdot \log(1/p_{\min}) \cdot \mathrm{OPT}_{\max}). \tag{1}$$

4. Results for Greedy Active Learning

The approximation factor guarantees cited above all inversely depend on $p_{\min}$, the smallest probability of any hypothesis in the given hypothesis class, according to the given distribution. Thus, if $p_{\min}$ is very small, the approximation factor is large, regardless of the true target hypothesis. We show that by slightly changing the policy of an approximately-greedy algorithm, we can achieve a better approximation factor whenever the true target hypothesis has a larger probability than $p_{\min}$. This can be done by allowing the algorithm to stop before it reaches a pure version space, that is before it can be certain of the correct labeling of the pool, and requiring that in this case, it would output the labeling which is most likely based on the current version space and the fixed probability distribution P. We say that $\mathcal{A}$ *outputs an approximate majority vote* if whenever V_T is pure enough, the algorithm outputs the majority vote on V_T. Formally, we define this as follows.

Definition 4 *An algorithm $\mathcal{A}$ outputs a β-approximate majority vote for $\beta \in (\frac{1}{2},1)$ if whenever there exists a labeling $Z : X \to \{\pm 1\}$ such that $\mathbb{P}_{h\sim P}[Z \Leftarrow h \mid h \in V_T] \geq \beta$, $\mathcal{A}$ outputs Z.*

In the following theorem we provide the target-dependent label complexity bound, which holds for any approximate greedy algorithm that outputs an approximate majority vote. We give here a sketch of the proof idea, the complete proof can be found in Appendix A.

Theorem 5 *Let $X = \{x_1,\ldots,x_m\}$. Let $\mathcal{H}$ be a hypothesis class, and let P be a distribution over $\mathcal{H}$. Suppose that $\mathcal{A}$ is α-approximately greedy with respect to P. Further suppose that it outputs a β-approximate majority vote. If $\mathcal{A}$ is executed with input (X,L,T) where $L \Leftarrow h \in \mathcal{H}$, then for all*

$$T \geq \alpha(2\ln(1/P(h)) + \ln(\frac{\beta}{1-\beta})) \cdot \mathrm{OPT}_{\max},$$

$\mathcal{A}$ outputs $L(1),\ldots,L(m)$.

Proof [Sketch] Fix a pool X. For any algorithm alg, denote by $V_t(\mathrm{alg},h)$ the version space induced by the first n labels it queries if the true labeling of the pool is consistent with h. Denote the average version space reduction of alg after t queries by

$$f_{\mathrm{avg}}(\mathrm{alg},t) = 1 - \mathbb{E}_{h\sim P}[P(V_t(\mathrm{alg},h))].$$

Golovin and Krause (2010) prove that since $\mathcal{A}$ is α-approximately greedy, for any pool-based algorithm alg, and for every $k,t \in \mathbb{N}$,

$$f_{\mathrm{avg}}(\mathcal{A},t) \geq f_{\mathrm{avg}}(\mathrm{alg},k)(1-\exp(-t/\alpha k)). \tag{2}$$

Let opt be an algorithm that achieves $\mathrm{OPT}_{\max}$. We show (see Appendix A) that for any hypothesis $h \in \mathcal{H}$ and any active learner alg,

$$f_{\mathrm{avg}}(\mathrm{opt},\mathrm{OPT}_{\max}) - f_{\mathrm{avg}}(\mathrm{alg},t) \geq P(h)(P(V_t(\mathrm{alg},h)) - P(h)).$$

Combining this with Equation (2) we conclude that if $\mathcal{A}$ is α-approximately greedy then

$$\frac{P(h)}{P(V_t(\mathcal{A},h))} \geq \frac{P(h)^2}{\exp(-\frac{t}{\alpha \mathrm{OPT}_{\max}}) + P(h)^2}.$$

This means that if $P(h)$ is large enough and we run an approximate greedy algorithm, then after a sufficient number of iterations, most of the remaining version space induces the correct labeling of the sample. Specifically, if $t \geq \alpha(2\ln(1/P(h)) + \ln(\frac{\beta}{1-\beta})) \cdot \mathrm{OPT}_{\max}$, then $P(h)/P(V_t(\mathcal{A},h)) \geq \beta$. Since $\mathcal{A}$ outputs a β-approximate majority labeling from $V_t(\mathcal{A},h)$, $\mathcal{A}$ returns the correct labeling. ■

When $P(h) \gg p_{\min}$, the bound in Theorem 5 is stronger than the guarantee in Equation (1), obtained by Golovin and Krause (2010). Note, however, that this bound depends on the probability of the target

hypothesis and thus is not known a-priori, unless additional assumptions are made. The margin assumption, which we discuss below, is an example for such a plausible assumption. Moreover, our experimental results indicate that even when such an apriori bound is not known, using a majority vote is preferable to selecting an arbitrary random hypothesis from an impure version space (see Figure 1 in Section 6.2).

Importantly, such an improved approximation factor *cannot be obtained* for a general approximate-greedy algorithm, even in a very simple setting. Thus, we can conclude that some algorithmic change is necessary. To show this, consider the setting of *thresholds on the line*. In this setting, the domain of examples is $[0,1]$, and the hypothesis class includes all the hypotheses defined by a threshold on $[0,1]$. Formally,

$$\mathcal{H}_{\text{line}} = \{h_c \mid c \in [0,1], h_c(x) = 1 \Leftrightarrow x \geq c\}.$$

Note that this setting is isomorphic to the case of homogeneous halfspaces with examples on a line in any Euclidean space of two or more dimensions.

Theorem 6 *Consider pool-based active learning on $\mathcal{H}_{\text{line}}$, and assume that P on $\mathcal{H}_{\text{line}}$ selects h_c by drawing the value c uniformly from $[0,1]$. For any $\alpha > 1$ there exists an α-approximately greedy algorithm $\mathcal{A}$ such that for any $m > 0$ there exists a pool $X \subseteq [0,1]$ of size m, and a threshold c such that $P(h_c) = 1/2$, while the label-complexity of $\mathcal{A}$ for $L \Leftarrow h_c$ is $\frac{m}{\lceil \log(m) \rceil} \cdot \text{OPT}_{\max}$.*

Proof For the hypothesis class $\mathcal{H}_{\text{line}}$, the possible version spaces after a partial run of an active learner are all of the form $[a,b] \subseteq [0,1]$.

First, it is easy to see that binary search on the pool can identify any hypothesis in $[0,1]$ using $\lceil \log(m) \rceil$ example, thus $\text{OPT}_{\max} = \lceil \log(m) \rceil$. Now, Consider an active learning algorithm that satisfies the following properties:

- If the current version space is $[a,b]$, it queries the smallest x that would still make the algorithm α-approximately greedy. Formally, it selects

$$x = \min\{x \in X \mid (x-a)(b-x) \geq \frac{1}{\alpha} \max_{\tilde{x} \in X \cap [a,b]} (\tilde{x}-a)(b-\tilde{x})\}.$$

- When the budget of queries is exhausted, if the version space is $[a,b]$, then the algorithm labels the points above a as positive and the rest as negative.

It is easy to see that this algorithm is α-approximately greedy, since in this problem $V^1_{t,x} \cdot V^{-1}_{t,x} = (x-a)(b-x)$ for all $x \in [a,b] = V_t$. Now for a given pool size $m \geq 2$, consider a pool of examples defined as follows. First, let $x_1 = 1$, $x_2 = 1/2$ and $x_3 = 0$. Second, for each $i \geq 3$, define x_{i+1} recursively as the solution to $(x_{i+1} - x_i)(1 - x_{i+1}) = \frac{1}{\alpha}(x_2 - x_i)(x_1 - x_2)$. Since $\alpha > 1$, it is easy to see by induction that for all $i \geq 3$, $x_{i+1} \in (x_i, x_2)$. Furthermore, suppose the true labeling is induced by $h_{3/4}$; Thus the only pool example with a positive label is x_1, and $P(h_{3/4}) = 1/2$. In this case, the algorithm we just defined will query all the pool examples $x_4, x_5, \ldots, x_m$ in order, and only then will it query x_2 and finally x_1. If stopped at any time $t \leq m-1$, it will label all the points that it has not queried yet as positive, thus if $t < m-1$ the output will be an erroneous labeling. Finally, note that the same holds for the pool $x_1, x_2, x_4, \ldots, x_m$ that does not include x_3, so the algorithm must query this entire pool to identify the correct labeling. ■

Interestingly, this theorem does not hold for $\alpha = 1$, that is for the exact greedy algorithm. This follows from Theorem 18, which we state and prove in Section 6.

So far we have considered a general hypothesis class. We now discuss the class of halfspaces in $\mathbb{R}^d$, denoted by $\mathcal{W}$ above. For simplicity, we will slightly overload notation and sometimes use w to denote the halfspace it determines. Every hypothesis in $\mathcal{W}$ can be described by a vector $w \in \mathbb{B}^d_1$, where $\mathbb{B}^d_1$ is the Euclidean unit ball, $\mathbb{B}^d_1 = \{w \in \mathbb{R}^d \mid \|w\| \leq 1\}$. We fix the distribution P to be the one that selects a vector w uniformly from $\mathbb{B}^d_1$. Our active learning algorithm for halfspaces, which is called ALuMA, is presented in Section 5. ALuMA receives as input an extra parameter $\delta \in (0,1)$, which serves as a measure of the desired confidence level. The following lemma, which we prove in Section 5, shows that ALuMA has the desired properties described above with high probability.

Lemma 7 *If ALuMA is executed with confidence δ, then with probability $1-\delta$ over its internal randomization, ALuMA is 4-approximately greedy and outputs a 2/3-approximate majority vote. Furthermore, ALuMA is polynomial in the pool size, the dimension, and $\log(1/\delta)$.*

Combining the above lemma with Theorem 5 we immediately obtain that ALuMA's label complexity is $O(\log(1/P(h)) \cdot \mathrm{OPT}_{\max})$. We can upper-bound $\log(1/P(h))$ using the familiar notion of *margin*: For any hypothesis $h \in \mathcal{W}$ defined by some $w \in \mathbb{B}_1^d$, let $\gamma(h)$ be the maximal margin of the labeling of X by h, namely $\gamma(h) = \max_{v:\|v\|=1} \min_{i\in[m]} h(x_i)\langle v, x_i\rangle / \|x_i\|$. We have the following lemma, which we prove in Appendix D:

Lemma 8 *For all $h \in \mathcal{W}$, $P(h) \geq \left(\frac{\gamma(h)}{2}\right)^d$.*

From Lemma 8 and Lemma 7, we obtain the following corollary, which provides a guarantee for ALuMA that depends on the margin of the target hypothesis.

Corollary 9 *Let $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{B}_1^d$, where $\mathbb{B}_1^d$ is the unit Euclidean ball of $\mathbb{R}^d$. Let $\delta \in (0,1)$ be a confidence parameter. Suppose that ALuMA is executed with input (X, L, T, δ), where $L \Leftarrow h \in \mathcal{W}$ and $T \geq 4(2d\ln(2/\gamma(h)) + \ln(2)) \cdot \mathrm{OPT}_{\max}$. Then, with probability of at least $1-\delta$ over ALuMA's own randomization, it outputs $L(1), \ldots, L(m)$.*

Note that ALuMA is allowed to use randomization, and it can fail to output the correct label with probability δ. In contrast, in the definition of $\mathrm{OPT}_{\max}$ we required that the optimal algorithm always succeeds, in effect making it deterministic. One may suggest that the approximation factor we achieve for ALuMA in Lemma 7 is due to this seeming advantage for ALuMA. We now show that this is not the case—the same approximation factor can be achieved when ALuMA and the optimal algorithm are allowed the same probability of failure. Let m be the size of the pool and let d be the dimension of the examples, and set $\delta_0 = \frac{1}{2m^d}$. Denote by $N_\delta(\mathcal{A}, h)$ the number of calls to L that $\mathcal{A}$ makes before outputting $(L(x_1), \ldots, L(x_m))$ with probability at least $1-\delta$, for $L \Leftarrow h$. Define $\mathrm{OPT}_{\delta_0} = \min_A \max_h N_{\delta_0}(A, h)$.

First, note that by setting $\delta = \delta_0$ in ALuMA, we get that $N_\delta(\mathrm{ALuMA}, h) \leq O(\log(1/P(h)) \cdot \mathrm{OPT}_{\max})$. Moreover, ALuMA with $\delta = \delta_0$ is polynomial in m and d (since it is polynomial in $\ln(1/\delta)$). Second, by Sauer's lemma there are at most m^d different possible labelings for the given pool. Thus by the union bound, there exists a fixed choice of the random bits used by an algorithm that achieves OPT_{δ_0}, that leads to the correct identification of the labeling for *all* possible labelings $L(1), \ldots, L(m)$. It follows that $\mathrm{OPT}_{\delta_0} = \mathrm{OPT}_{\max}$. Therefore the same factor of approximation can be achieved for ALuMA with $\delta = \delta_0$, compared to OPT_{δ_0}.

Our result for ALuMA provides a target-dependent approximation factor guarantee, depending on the margin of the target hypothesis. We can also consider the minimal possible margin, $\gamma = \min_{h\in\mathcal{W}} \gamma(h)$, and deduce from Corollary 9, or from the results of Golovin and Krause (2010), a uniform approximation factor of $O(d\log(1/\gamma))$. How small can γ be? The following result bounds this minimal margin from below under the reasonable assumption that the examples are represented by numbers of a finite accuracy.

Lemma 10 *Let $c > 0$ be such that $1/c$ is an integer and suppose that $X \subset \{-1, -1+c, \ldots, 1-c, 1\}^d$. Then, $\min_{h\in\mathcal{W}} \gamma(h) \geq (c/\sqrt{d})^{d+2}$.*

The proof, given in Appendix D, is an adaptation of a classic result due to Muroga et al. (1961). We conclude that under this assumption for halfspaces, $p_{\min} = \Omega((c/d)^{d^2})$, and deduce an approximation factor of $d^2\log(d/c)$ for the worst-case label complexity of ALuMA. The exponential dependence of the minimal margin on d here is necessary; as shown in Håstad (1994), the minimal margin can indeed be exponentially small, even if the points are taken only from $\{\pm 1\}^d$.

We also derive a lower bound, showing that the dependence of our bounds on γ or on c is necessary. Whether the dependence on d is also necessary is an open question for future work.

Theorem 11 *For any $\gamma \in (0, 1/8)$, there exists a pool $X \subseteq \mathbb{B}_1^2 \cap \{-1, 1+c, \ldots, 1-c, 1\}^2$ for $c = \Theta(\gamma)$, and a target hypothesis $h^* \in \mathcal{W}$ for which $\gamma(h^*) = \Omega(\gamma)$, such that there exists an exact greedy algorithm that requires $\Omega(\ln(1/\gamma)) = \Omega(\ln(1/c))$ labels in order to output a correct majority vote, while the optimal algorithm requires only $O(\log(\log(1/\gamma)))$ queries.*

The proof of Theorem 11 is provided in Appendix D. In the next section we describe the ALuMA algorithm in detail.

5. The ALuMA Algorithm

We now describe our algorithm, listed below as Alg. 1, and explain why Lemma 7 holds. We name the algorithm *Active Learning under a Margin Assumption* or ALuMA. Its inputs are the unlabeled sample X, the labeling oracle L, the maximal allowed number of label queries T, and the desired confidence $\delta \in (0,1)$. It returns the labels of all the examples in X.

As we discussed earlier, in each iteration, we wish to choose among the instances in the pool, the instance whose label would lead to the maximal (expected) reduction in the version space. Denote by I_t the set of indices corresponding to the elements in the pool whose label was not queried yet ($I_0 = [m]$). Then, in round t, we wish to find

$$k = \operatorname*{argmax}_{i \in I_t} P(V^1_{t,x_i}) \cdot P(V^{-1}_{t,x_i}). \tag{3}$$

Recall we take P to be uniform over $\mathcal{W}$, the class of homogenous half-spaces in $\mathbb{R}^d$. In this case, the probability of a version space is equivalent to its volume, up to constant factors. Therefore, in order to be able to solve Equation (3), we need to calculate the volumes of the sets $V^1_{t,x}$ and $V^{-1}_{t,x}$ for every element x in the pool. Both of these sets are convex sets obtained by intersecting the unit ball with halfspaces. The problem of calculating the volume of such convex sets in $\mathbb{R}^d$ is #P-hard if d is not fixed (Brightwell and Winkler, 1991). In many learning applications d is large, therefore, indeed d should not be taken as fixed. Moreover, deterministically approximating the volume is NP-hard in the general case (Matoušek, 2002). Luckily, it is possible to approximate this volume using randomization. Specifically, in Kannan et al. (1997) a randomized algorithm with the following guarantees is provided, where $\mathrm{Vol}(K)$ denotes the volume of the set K.

Lemma 12 *Let $K \subseteq \mathbb{R}^d$ be a convex body with an efficient separation oracle. There exists a randomized algorithm, such that given $\varepsilon, \delta > 0$, with probability at least $1-\delta$ the algorithm returns a non-negative number Γ such that $(1-\varepsilon)\Gamma < \mathrm{Vol}(K) < (1+\varepsilon)\Gamma$. The running time of the algorithm is polynomial in $d, 1/\varepsilon, \ln(1/\delta)$.*

We denote an execution of this algorithm on a convex body K by $\Gamma \leftarrow \mathrm{VolEst}(K, \varepsilon, \delta)$. The algorithm is polynomial in $d, 1/\varepsilon, \ln(1/\delta)$. ALuMA uses this algorithm to estimate $P(V^1_{t,x})$ and $P(V^{-1}_{t,x})$ with sufficient accuracy. We denote these approximations by $\hat{v}_{x,1}$ and $\hat{v}_{x,-1}$ respectively. Using the constants in ALuMA, we can show the following.

Lemma 13 *With probability at least $1-\delta/2$, Alg. 1 is 4-approximately greedy.*

Proof Fix some $t \in [T]$. Let $k \in I_t$ be the index chosen by ALuMA. Let k^* be the index corresponding to the value of Equation (3). Since ALuMA performs at most $2m$ approximations in each round, we obtain by Lemma 12 and the union bound that with probability at least $1 - \frac{\delta}{2T}$, for each $i \in I_t$ and each $j \in \{-1, 1\}$,

$$\hat{v}_{x_i,j} \in \left(\frac{2}{3}\mathrm{Vol}(V^j_{t,x_i}), \frac{4}{3}\mathrm{Vol}(V^j_{t,x_i}) \right).$$

In addition, $\hat{v}_{x_k,1} \cdot \hat{v}_{x_k,-1} \geq \hat{v}_{x_{k^*},1} \cdot \hat{v}_{x_{k^*},-1}$. Hence, with probability at least $1 - \frac{\delta}{2T}$,

$$\frac{16}{9}\mathrm{Vol}(V^{-1}_{t,x_k}) \cdot \mathrm{Vol}(V^1_{t,x_k}) \geq \frac{4}{9}\mathrm{Vol}(V^{-1}_{t,x_{k^*}}) \cdot \mathrm{Vol}(V^1_{t,x_{k^*}}).$$

Applying the union bound over T iteration completes our proof. ■

After T iterations, ALuMA needs to output the majority vote of a version space that has a high enough purity level. To output an approximate majority vote from the final version space V, we would like to uniformly draw several hypotheses from V and label X according to a majority vote over these hypotheses. The

Algorithm 1 The **ALuMA** algorithm

Input: $X = \{x_1, \ldots, x_m\}$, $L : [m] \to \{-1, 1\}$, T, δ
$I_1 \leftarrow [m]$, $V_1 \leftarrow \mathbb{B}_1^d$
for $t = 1$ to T **do**
    $\forall i \in I_t, j \in \{\pm 1\}$, do $\hat{v}_{x_i,j} \leftarrow \text{VolEst}(V_{t,x_i}^j, \frac{1}{3}, \frac{\delta}{4mT})$
    Select $i_t \in \text{argmax}_{i \in I_t} (\hat{v}_{x_i,1} \cdot \hat{v}_{x_i,-1})$
    $I_{t+1} \leftarrow I_t \setminus \{i_t\}$
    Request $y = L(i_t)$
    $V_{t+1} \leftarrow V_t \cap \{w : y\langle w, x_{i_t}\rangle > 0\}$
end for
$M \leftarrow \lceil 72 \ln(2/\delta) \rceil$.
Draw $w_1, \ldots, w_M$ $\frac{1}{12}$-uniformly from V_{T+1}.
For each x_i return the label $y_i = \text{sgn}\left(\sum_{j=1}^M \text{sgn}(\langle w_j, x_i \rangle)\right)$.

task of uniformly drawing hyphteses from V can be approximated using the hit-and-run algorithm (Lovász, 1999). The hit-and-run algorithm efficiently draws a random sample from a convex body K according to a distribution which is close in total variation distance to the uniform distribution over K. Formally, The following definition parametrizes the closeness of a distribution to the uniform distribution:

Definition 14 *Let $K \subseteq \mathbb{R}^d$ be a convex body with an efficient separation oracle, and let τ be a distribution over K. τ is* λ-uniform *if* $\sup_A |\tau(A) - P(A)/P(K)| \leq \lambda$, *where the supremum is over all measurable subsets of K.*

The hit-and-run algorithm draws a sample from a λ-uniform distribution in time $\tilde{O}(d^3/\lambda^2)$. The next lemma shows that using the hit-and-run as suggested above indeed produces a majority vote classification.

Lemma 15 *ALuMA outputs a $2/3$-approximate majority vote with probability at least $1 - \delta/2$.*

Proof Assume that there exists a labeling $Z : X \to \{\pm 1\}$ such that $\mathbb{P}_{h \sim P}[Z \Leftarrow h \mid h \in V_{T+1}] \geq 2/3$. In step 11 of ALuMA, $M \geq 72 \ln(2/\delta)$ hypotheses are drawn $\frac{1}{12}$-uniformly at random from V_t. Therefore each hypothesis $h_i \in V_{T+1}$ is consistent with Z with probability at least $\frac{7}{12}$. By Hoeffding's inequality,

$$\mathbb{P}\left[\frac{1}{M}\sum_{i=1}^M I[h_i \in V(h|_X)] \leq \frac{1}{2}\right] \leq \exp(-M/72) = \frac{\delta}{2}.$$

Therefore, with probability at least $1 - \delta/2$, ALuMA outputs a $2/3$-approximate majority vote. ■

We can now prove Lemma 7.

Proof (Of Lemma 7) Lemma 13 and Lemma 15 above prove the first two parts of the lemma. We only have left to analyze the time complexity of ALuMA. In each iteration, the cost of ALuMA is dominated by the cost of performing at most $2m$ volume approximation, each of which costs $O(d^5 \ln(1/\delta))$. As we discussed above, implementing the majority vote costs polynomial time in d and $\ln(1/\delta)$. Overall, the runtime of ALuMA is polynomial in m (which upper bounds T), d and $\log(1/\delta)$. ■

5.1 A Simpler Implementation of ALuMA

The ALuMA algorithm described in Alg. 1 uses $O(Tm)$ volume estimations as a black-box procedure, where T is the budget of labels and m is the pool size. The complexity of each application of the volume estimation

procedure is $\tilde{O}(d^5)$ where d is the dimension. Thus the overall complexity of the algorithm is $\tilde{O}(Tmd^5)$. This complexity can be somewhat improved under some "luckiness" conditions.

The volume estimation procedure uses λ-uniform sampling based on hit-and-run as its core procedure. Instead, we can use hit-and-run directly as follows: At each iteration of ALuMA, instead of step 4, perform the following procedure:

Algorithm 2 Estimation Procedure

1: Input: $\lambda \in (0, \frac{1}{24}), V_t, I_t$
2: $k \leftarrow \frac{\ln(2Nm/\delta)}{2\lambda^2}$
3: Sample $h_1, \ldots, h_k \in V_t$ λ-uniformly.
4: $\forall i \in I_t, j \in \{-1,+1\}$, $\hat{v}_{x_i,j} \leftarrow \frac{1}{k}|\{i \mid h_i(x_i) = j\}|$.

The complexity of ALuMA when using this procedure is $\tilde{O}(T(d^3/\lambda^4 + m/\lambda^2))$, which is better than the complexity of the full Alg. 1 for a constant λ. An additional practical benefit of this alternative estimation procedure is that when implementing, it is easy to limit the actual computation time used in the implementation by running the procedure with a smaller number k and a smaller number of hit-and-run mixing iterations.[2] This provides a natural trade-off between computation time and labeling costs.

The following theorem shows that under mild conditions, using the estimation procedure listed in Alg. 2 also results in an approximately greedy algorithm, as does the original implementation of ALuMA.

Theorem 16 *If for each iteration t of the algorithm, the greedy choice x^* satisfies*

$$\forall j \in \{-1,+1\}, \quad \mathbb{P}[h(x^*) = j \mid h \in V_t] \geq 4\sqrt{\lambda}$$

then ALuMA with the estimation procedure is a 2-approximate greedy algorithm. Moreover, it is possible to efficiently verify that this condition holds while running the algorithm.

Proof Fix the iteration t, and denote $p_{x,1} = P(V^1_{t,x})/P(V_t)$ and $p_{x,-1} = P(V^1_{t,x})/P(V_t)$. Note that $p_{x,1} + p_{x,-1} = 1$. Since $h_1, \ldots, h_k$ are sampled λ-uniformly from the version space, we have

$$\forall i \in [k], |\mathbb{P}[h_i \in V^j_{t,x}] - p_{x,j}| \leq \lambda. \tag{4}$$

In addition, by Hoeffding's inequality and a union bound over the examples in the pool and the iterations of the algorithm,

$$\mathbb{P}[\exists x, |\hat{v}_{x_i,j} - \mathbb{P}[h_i \in V^j_{t,x}]| \geq \lambda] \leq 2m\exp(-2k\lambda^2). \tag{5}$$

From Alg. 2 we have $k = \frac{\ln(2m/\delta)}{2\lambda^2}$. Combining this with Equation (4) and Equation (5) we get that

$$\mathbb{P}[\exists x, |\hat{v}_{x_i,j} - p_{x_i,j}]| \geq 2\lambda] \leq \delta.$$

The greedy choice for this iteration is

$$x^* \in \operatorname*{argmax}_{x \in X} \Delta(h|_X, x) = \operatorname*{argmax}_{x \in X} (p_{x,1} p_{x,-1}).$$

By the assumption in the theorem, $p_{x^*,j} \geq 4\sqrt{\lambda}$ for $j \in \{-1,+1\}$. Since $\lambda \in (0, \frac{1}{64})$, we have $\lambda \leq \sqrt{\lambda}/8$. Therefore $p_{x^*,j} - 2\lambda \geq 4\sqrt{\lambda} - \sqrt{\lambda}/4 \geq \sqrt{10\lambda}$. Therefore

$$\hat{v}_{x^*,1}\hat{v}_{x^*,-1} \geq (p_{x^*,1} - 2\lambda)(p_{x^*,-1} - 2\lambda) \geq 10\lambda. \tag{6}$$

2. Gilad-Bachrach et al. (2005) report that the actual mixing time of hit-and-run is much faster than the one guaranteed by the theoretical bounds, and we have observed a similar phenomenon in our experiments.

Let $\tilde{x} = \text{argmax}(\hat{v}_{x,-1}\hat{v}_{x,+1})$ be the query selected by ALuMA using Alg. 2. Then

$$\hat{v}_{x^*,-1}\hat{v}_{x^*,+1} \leq \hat{v}_{\tilde{x},-1}\hat{v}_{\tilde{x},+1} \leq (p_{\tilde{x},1}+2\lambda)(p_{\tilde{x},-1}+2\lambda) \leq p_{\tilde{x},1}p_{\tilde{x},-1}+4\lambda.$$

Where in the last inequality we used the facts that $p_{\tilde{x},1}+p_{\tilde{x},-1}=1$ and $4\lambda^2 \leq 2\lambda$. On the other hand, by Equation (6)

$$\hat{v}_{x^*,-1}\hat{v}_{x^*,+1} \geq 5\lambda+\frac{1}{2}\hat{v}_{x^*,-1}\hat{v}_{x^*,+1} \geq 5\lambda+\frac{1}{2}(p_{x^*,-1}-2\lambda)(p_{x^*,-1}-2\lambda) \geq 4\lambda+\frac{1}{2}p_{x^*,-1}p_{x^*,-1}.$$

Combining the two inequalities for $\hat{v}_{x^*,-1}\hat{v}_{x^*,+1}$ it follows that $p_{\tilde{x},1}p_{\tilde{x},-1} \geq \frac{1}{2}p_{x^*,-1}p_{x^*,-1}$, thus this is a 2-approximately greedy algorithm.

To verify that the assumption holds at each iteration of the algorithm, note that for all $x = x_i$ such that $i \in I_t$

$$p_{x,-1}p_{x,+1} \geq (\hat{v}_{x,-1}-2\lambda)(\hat{v}_{x,+1}-2\lambda) \geq \hat{v}_{x,-1}\hat{v}_{x,+1}-2\lambda.$$

therefore it suffices to check that for all $x = x_i$ such that $i \in I_t$ $\hat{v}_{x,-1}\hat{v}_{x,+1} \geq 4\sqrt{\lambda}+2\lambda$. ■

The condition added in this theorem is that the best example in each iteration should induce a fairly balanced partition of the current version space. In our experiments we noticed that this is generally the case in practice. Moreover, the theorem shows that it is possible to verify that the condition holds while running the algorithm. Thus, the estimation procedure can easily be augmented with an additional verification step at the beginning of each iteration. On iterations that fail the verification, the algorithm will use the original black-box volume estimation procedure. We have used this simpler implementation in our experiments, which are reported below.

5.2 Handling Non-Separable Data and Kernel Representations

If the data pool X is not separable, but a small upper bound on the total hinge-loss of the best separator can be assumed, then ALuMA can be applied after a preprocessing step, which we describe in detail below. This preprocessing step maps the points in X to a set of points in a higher dimension, which are separable using the original labels of X. The dimensionality depends on the margin and on the bound on the total hinge-loss of the original representation. The preprocessing step also supports kernel representations, so that the original X can be represented by a kernel matrix as well. Applying ALuMA after this preprocessing steps results in an approximately optimal label complexity, however OPT_{max} here is measured with respect to the new representation.

While some of the transformations we employ in the preprocessing step have been discussed before in other contexts (see, e.g., Balcan et al., 2006b), we describe and analyze the full procedure here for completeness. The preprocessing step is composed of two simple transformations. In the first transformation each example $x_i \in X$ is mapped to an example in dimension $d+m$, defined by $x_i' = (ax_i; \sqrt{1-a^2}\cdot e_i)$, where e_i is the i'th vector of the natural basis of $\mathbb{R}^m$ and $a > 0$ is a scalar that will be defined below. Thus the first d coordinates of x_i' hold the original vector times a, the rest of the coordinates are zero,except for $x_i'[d+i] = \sqrt{1-a^2}$. This mapping guarantees that the set $X' = (x_1', \ldots, x_m')$ is separable with the same labels as those of X, and with a margin that depends on the cumulative squared-hinge-loss of the data.

In the second transformation, a Johnson-Lindenstrauss random projection (Johnson and Lindenstrauss, 1984; Bourgain, 1985) is applied to X', thus producing a new set of points $\bar{X} = (\bar{x}_1, \ldots, \bar{x}_m)$ in a different dimension $\mathbb{R}^k$, where k depends on the original margin and on the amount of margin error. With high probability, the new set of points will be separable with a margin that also depends on the original margin and on the amount of margin error. If the input data is provided not as vectors in $\mathbb{R}^d$ but via a kernel matrix, then a simple decomposition is performed before the preprocessing begins.

The full preprocessing procedure is listed below as Alg. 3. The first input to the algorithm is the data for preprocessing, given as $X \subseteq \mathbb{R}^d$ or as a kernel matrix $K \in \mathbb{R}^{m\times m}$. The other inputs are γ—a margin parameter, H—an upper bound on the margin error relative to γ, and δ, which is the required confidence.

Algorithm 3 Preprocessing

Input: $X = \{x_1, \ldots, x_m\} \in \mathbb{R}^d$ or $K \in \mathbb{R}^{m \times m}$, γ, H, δ
if input data is a kernel matrix K **then**
    Find $U \in \mathbb{R}^{m \times m}$ such that $K = UU^T$
    $\forall i \in [m], x_i \leftarrow$ row i of U
    $d \leftarrow m$
end if
$a \leftarrow \sqrt{\frac{1}{1+\sqrt{H}}}$
$\forall i \in [m], x_i' \leftarrow (ax_i; \sqrt{1-a^2} \cdot e_i)$
$k \leftarrow O\left(\frac{(H+1)\ln(m/\delta)}{\gamma^2}\right)$
$M \leftarrow$ a random $\{\pm 1\}$ matrix of dimension $k \times (d+m)$
for $i \in [m]$ **do**
    $\bar{x}_i \leftarrow M x_i'$
end for
Return $(\bar{x}_1, \ldots, \bar{x}_m)$.

After the preprocessing step, $\bar{X}$ is used as input to ALuMA, which then returns a set of labels for the examples in $\bar{X}$. These are also the labels of the examples in the original X. To retrieve a halfspace for X with the least margin error, any passive learning algorithm can be applied to the resulting labeled sample. The full active learning procedure is described in Alg. 4.

Note that if ALuMA returns the correct labels for the sample, the usual generalization bounds for passive supervised learning can be used to bound the true error of the returned separator w. In particular, we can apply the support vector machine algorithm (SVM) and rely on generalization bounds for SVM.

Algorithm 4 Active Learning

Input: $X = \{x_1, \ldots, x_m\}$ or $K \in \mathbb{R}^{m \times m}$, $L : [m] \to \{-1, 1\}$, N, γ, H, δ
if input has X **then**
    Get $\bar{X}$ by running Alg. 3 with input $X, \gamma, H, \delta/2$.
else
    Get $\bar{X}$ by running Alg. 3 with input $K, \gamma, H, \delta/2$.
end if
Get $(y_1, \ldots, y_m)$ by running ALuMA with input $\bar{X}$, L, N, $\delta/2$.
Get $w \in \mathbb{R}^d$ by running SVM on the labeled sample $\{(x_1, y_1), \ldots, (x_m, y_m)\}$.
Return w.

The result of these transformations are summarized in the following theorem.

Theorem 17 *Let $X = \{x_1, \ldots, x_m\} \subseteq B$, where B is the unit ball in some Hilbert space. Let $H \geq 0$ and $\gamma > 0$, and assume there exists a $w^* \in B$ such that*

$$H \geq \sum_{i=1}^{m} \max(0, \gamma - L(i)\langle w^*, x_i \rangle)^2.$$

Let $\delta \in (0,1)$ be a confidence parameter. There exists an algorithm that receives X as vectors in $\mathbb{R}^d$ or as a kernel matrix $K \in \mathbb{R}^{m \times m}$, and input parameters γ and H, and outputs a set $\bar{X} = \{\bar{x}_1, \ldots, \bar{x}_m\} \subseteq \mathbb{R}^k$, such that

1. $k = O\left(\frac{(H+1)\ln(m/\delta)}{\gamma^2}\right)$,
2. *With probability $1 - \delta$, $\bar{X} \subseteq \mathbb{B}_1^k$ and $(\bar{X}, L)$ is separable with a margin $\frac{\gamma}{2+2\sqrt{H}}$.*

3. *The run-time of the algorithm is polynomial in $d, m, 1/\gamma, \ln(1/\delta)$ if x_i are represented as vectors in d, and is polynomial in $m, 1/\gamma, \ln(1/\delta)$ if x_i are represented by a kernel matrix.*

The proof of Theorem 17 can be found in Appendix B. In Section 6.2 we demonstrate that in practice, this procedure provides good label complexity results on real data sets. Investigating the relationship between $\mathrm{OPT}_{\max}$ in the new representation and $\mathrm{OPT}_{\max}$ in the original representation is an important question for future work.

6. Other Approaches: A Theoretical and Empirical Comparison

We now compare the effectiveness of the approach implemented by ALuMA to other active learning strategies. ALuMA can be characterized by two properties: (1) its "objective" is to reduce the volume of the version space and (2) at each iteration, it aggressively selects an example from the pool so as to (approximately) minimize its objective as much as possible (in a greedy sense). We discuss the implications of these properties by comparing to other strategies. Property (1) is contrasted with strategies that focus on increasing the number of examples whose label is known. Property (2) is contrasted with strategies which are "mellow", in that their criterion for querying examples is softer.

Much research has been devoted to the challenge of obtaining a substantial guaranteed improvement of label complexity over regular "passive" learning for halfspaces in $\mathbb{R}^d$. Examples (for the realizable case) include the Query By Committee (QBC) algorithm (Seung et al., 1992; Freund et al., 1997), the CAL algorithm (Cohn et al., 1994), and the Active Perceptron (Dasgupta et al., 2005). These algorithms are not "pool-based" but rather use "selective-sampling": they sample one example at each iteration, and immediately decide whether to ask for its label. Out of these algorithms, CAL is the most mellow, since it queries any example whose label is yet undetermined by the version space. Its "objective" can be described as reducing the number of examples which are labeled incorrectly, since it has been shown to do so in many cases (Hanneke, 2007, 2011; Friedman, 2009). QBC and the active perceptron are less mellow. Their "objective" is similar to that of ALuMA since they decide on examples to query based on geometric considerations.

In Section 6.1 we discuss the theoretical advantages and disadvantages of different strategies, by considering some interesting cases from a theoretical perspective. In Section 6.2 we report an empirical comparison of several algorithms and discuss our conclusions.

6.1 Theoretical Comparison

The label complexity of the algorithms mentioned above is usually analyzed in the PAC setting, thus we translate our guarantees into the PAC setting as well for the sake of comparison. We define the (ε, m, D)-label complexity of an active learning algorithm to be the number of *label queries* that are required in order to guarantee that given a sample of m unlabeled examples drawn from D, the error of the learned classifier will be at most ε (with probability of at least $1-\delta$ over the choice of sample). A pool-based active learner can be used to learn a classifier in the PAC model by first sampling a pool of m unlabeled examples from D, then applying the pool-based active learner to this pool, and finally running a standard passive learner on the labeled pool to obtain a classifier. For the class of halfspaces, if we sample an unlabeled pool of $m = \tilde{\Omega}(d/\varepsilon)$ examples, then the learned classifier will have an error of at most ε (with high probability over the choice of the pool).

To demonstrate the effect of the first property discussed above, consider again the simple case of thresholds on the line defined in Section 4. Compare two greedy pool-based active learners for $\mathcal{H}_{\text{line}}$: The first follows a binary search procedure, greedily selecting the example that increases the number of known labels the most. Such an algorithm requires $\lceil \log(m) \rceil$ queries to identify the correct labeling of the pool. The second algorithm queries the example that splits the version space as evenly as possible. Theorem 5 implies a label complexity of $O(\log(m)\log(1/\gamma(h)))$ for such an algorithm, since $\mathrm{OPT}_{\max} = \lceil \log(m) \rceil$. However, a better result holds for this simple case:

Theorem 18 *In the problem of thresholds on the line, for any pool with labeling L, the exact greedy algorithm requires at most $O(\log(1/\gamma(h)))$ labels. This is also the label complexity of any approximate greedy algorithm that outputs a majority vote.*

Proof First, assume that the algorithm is exactly greedy. A version space for $\mathcal{H}_{\text{line}}$ is described by a segment in $[a,b] \subseteq [0,1]$, and a query at point α results in a new version space, $[a,\alpha]$ or $[\alpha,b]$, depending on the label. We now show that for every version space $[a,b]$, at most two greedy queries suffice to either reduce the size of the version space by a factor of at least $2/3$, or to determine the labels of all the points in the pool.

Assume for simplicity that the version space is $[0,1]$, and denote the pool of examples in the version space by X. Assume w.l.o.g. that the greedy algorithm now queries $\alpha \leq \frac{1}{2}$. If $\alpha > 1/3$, then any answer to the query will reduce the version space size to less than $2/3$. Thus assume that $\alpha \leq 1/3$. If the query answer results in the version space $[0,\alpha)$ then we are done since this version space is smaller than $2/3$. We are left with the case that the version space after querying α is $[\alpha,1]$. Since the algorithm is greedy, it follows that for $\beta = \min\{x \in X \mid x \geq \alpha\}$, we have $\beta \geq 1-\alpha$: this is because if there was a point $\beta \in (\alpha, 1-\alpha)$, it would cut the version space more evenly than α, in contradiction to the greedy choice of α. Note further that $(\alpha, 1-\alpha)$ is larger than $[1-\alpha,1]$ since $\alpha \leq 1/3$. Therefore, the most balanced choice for the greedy algorithm is β. If the query answer for β cuts the version space to $(\beta,1]$ then we are done, since $1-\beta \leq \alpha \leq 1/3$. Otherwise, the query answer leaves us with the version space (α,β). This version space includes no more pool points, by the definition of β. Thus in this case the algorithm has determined the labels of all points.

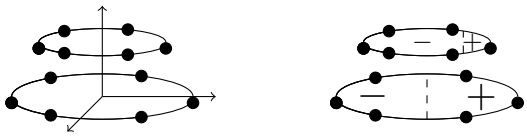
It follows that if the algorithm runs at least t iterations, then the size of the version space after t iterations is at most $(2/3)^{t/2}$. If the true labeling has a margin of γ, we conclude that $(2/3)^{t/2} \geq \gamma$, thus $t \leq O(\log(1/\gamma))$.

A similar argument can be carried for ALuMA, using a smaller bound on α and more iterations due to the approximation, and noting that if the correct answer is in $(\alpha, 1-\alpha)$ then a majority vote over thresholds drawn randomly from the version space will label the examples correctly. ■

Comparing the $\lceil \log(m) \rceil$ guarantee of the first algorithm to the $\log(1/\gamma(h))$ guarantee of the second, we reach the (unsurprising) conclusion, that the first algorithm is preferable when the true labeling has a small margin, while the second is preferable when the true labeling has a large margin. This simple example accentuates the implications of selecting the volume of the version space as an objective. A similar implication can be derived by considering the PAC setting, replacing the binary-search algorithm with CAL, and letting $m = \tilde{\Theta}(1/\varepsilon)$. On the single-dimensional line, CAL achieves a label-complexity of $O(\log(1/\varepsilon)) = O(\log(m))$, similarly to the binary search strategy we described. Thus when ε is large compared to $\gamma(h)$, CAL is better than being greedy on the volume, and the opposite holds when the condition is reversed. QBC will behave similarly to ALuMA in this setting.

To demonstrate the effect of the second property described above—being aggressive versus being mellow, we consider the following example, adapted slightly from Dasgupta (2006).

Example 19 *Consider two circles parallel to the (x,y) plane in $\mathbb{R}^3$, one at the origin and one slightly above it. For a given ε, fix $2/\varepsilon$ points that are evenly distributed on the top circle, and $2/\varepsilon$ points at the same angles on the bottom circle (see left illustration below). The distribution D_ε is an uneven mix of a uniform distribution over the points on the top circle and one over the points of the bottom circle: The top circle is given a much higher probability. All homogeneous separators label half of the bottom circle positively, but an unknown part of the top circle (see right illustration). The bottom points can be very helpful in finding the correct separator fast, but their probability is low.*



Dasgupta has demonstrated via this example that active learning can gain in label complexity from having significantly more unlabeled data. The following theorem shows that the aggressive strategy employed by

ALuMA indeed achieves an exponential improvement when there are more unlabeled samples. In many applications, unlabeled examples are virtually free to sample, thus it can be worthwhile to allow the active learner to sample more examples than the passive sample complexity and use an aggressive strategy.[3] In contrast, the mellow strategy of CAL does not significantly improve over passive learning in this case. We note that these results hold for any selective-sampling method that guarantees an error rate similar to passive ERM given the same sample size. This falls in line with the observation of Balcan et al. (2007), that in some cases a more aggressive approach is preferable.

Theorem 20 *For all small enough* $\varepsilon \in (0,1)$ *the distribution* D_ε *in Example 19 satisfies*

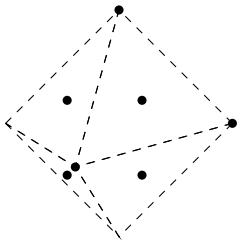
1. *For* $m = O(1/\varepsilon)$*, the* $(\varepsilon, m, D_\varepsilon)$*-label complexity of* any active learner *is* $\Omega(1/\varepsilon)$.
2. *For* $m = \Omega(\log^2(1/\varepsilon)/\varepsilon^2)$*, the* $(\varepsilon, m, D_\varepsilon)$*-label complexity of ALuMA is* $O(\log^2(1/\varepsilon))$.
3. *For* any value *of m, the* $(\varepsilon, m, D_\varepsilon)$*-label complexity of CAL is* $\Omega(1/\varepsilon)$.

The proof of Theorem 20 is provided in Appendix C. The example above demonstrated that more unlabeled examples can help ALuMA use less labels, whereas they do not help CAL. In fact, in some cases the label complexity of CAL can be significantly worse than that of the optimal algorithm, even when both CAL and the optimal algorithm have access to all the points in the support of the distribution. This is demonstrated in the following example. Note that in this example, a passive learner also requires access to all the points in the support of the distribution, thus CAL, passive learning, and optimal active learning all require the same size of a random unlabeled pool.

Example 21 *Consider a distribution in* $\mathbb{R}^d$ *that is supported by two types of points on an octahedron (see an illustration for* $\mathbb{R}^3$ *below).*

1. *Vertices:* $\{e_1, \ldots, e_d\}$.
2. *Face centers:* z/d *for* $z \in \{-1,+1\}^d$.

Consider the hypothesis class $\mathcal{W} = \{x \mapsto \operatorname{sgn}(\langle x, w\rangle - 1 + \frac{1}{d}) \mid w \in \{-1,+1\}^d\}$. *Each hypothesis in* $\mathcal{W}$*, defined by some* $w \in \{-1,+1\}^d$*, classifies at most* $d+1$ *data points as positive: these are the vertices* e_i *for* i *such that* $w[i] = +1$*, and the face center* w/d.



Theorem 22 *Consider Example 21 for* $d \geq 3$*, and assume that the pool of examples includes the entire support of the distribution. There is an efficient algorithm that finds the correct hypothesis from* $\mathcal{W}$ *with at most d labels. On the other hand, with probability at least* $\frac{1}{e}$ *over the randomization of the sample, CAL uses at least* $\frac{2^d+d}{2d+3}$ *labels to find the correct separator.*

3. In the limit of an infinite number of unlabeled examples, if the distribution has a non-zero support on the entire domain, the pool-based setting becomes identical to the setting of membership queries (Angluin, 1988). In contrast, we are interested in finite samples.

Proof First, it is easy to see that if $h^* \in \mathcal{W}$ is the correct hypothesis, then

$$w = (h^*(e_1), \ldots, h^*(e_d)).$$

Thus, it suffices to query the d vertices to discover the true w.

We now show that the number of queries CAL asks until finding the correct separator is exponential in d. CAL inspects the unlabeled examples sequentially, and queries any example whose label cannot be inferred from previous labels. Consider some run of CAL (determined by the random ordering of the sample). Assume w.l.o.g. that each data point appears once in the sample. Let S be the set that includes the positive face center and all the vertices. Note that CAL cannot terminate before either querying all the $2^d - 1$ negative face centers, or querying at least one example from S. Moreover, CAL will query all the face centers it encounters before encountering the first example from S. At each iteration t before encountering an example from S, there is a probability of $\frac{d+1}{2^d+d-t}$ that the next example is from S. Therefore, the probability that the first $T = \frac{2^d+d}{2d+3}$ examples are not from S is

$$\prod_{t=0}^{T-1}\left(1 - \frac{d+1}{2^d+d-t}\right) \geq \left(1 - \frac{d+1}{2^d+d-T}\right)^T \geq e^{-2T\frac{d+1}{2^d+d-T}} = e^{\frac{-2(d+1)}{\frac{2^d+d}{T}-1}} = \frac{1}{e}\,,$$

where in the second equality we used $1 - a \geq \exp(-2a)$ which holds for all $a \in [0, \frac{1}{2}]$. Therefore, with probability at least $\frac{1}{e}$ the number of queries is at least $\frac{2^d+d}{2d+3}$. ■

These examples show that in some cases an aggressive approach is preferable to a mellow approach such as employed by CAL. At the same time, it should be noted that CAL has a guaranteed label complexity for cases for which ALuMA currently has none. Its label complexity is bounded by $\tilde{O}(d\theta\log(1/\varepsilon))$, where θ is the disagreement coefficient, a quantity that depends on the distribution and the target hypothesis (Hanneke, 2007, 2011). Specifically, if D is uniform over a sphere centered at the origin, then for all target hypotheses $\theta = \Theta(\sqrt{d})$. Thus CAL achieves an exponential improvement over passive learning for this canonical example. We do not have a similar analysis for ALuMA for the case of a uniform distribution.

6.2 Empirical Comparison

We carried out an empirical comparison between the algorithms discussed above. Our goal is twofold: First, to evaluate ALuMA in practice, and second, to compare the performance of aggressive strategies compared to mellow strategies. The aggressive strategies are represented in this evaluation by ALuMA and one of the heuristics proposed by Tong and Koller (2002). The mellow strategy is represented by CAL. QBC represents a middle-ground between aggressive and mellow. We also compare to a passive ERM algorithm—one that uses random labeled examples. We evaluated the algorithms over synthetic and real data sets and compared their label complexity performance.

Our implementation of ALuMA uses hit-and-run samples instead of full-blown volume estimation, as described in Section 5.1. QBC is also implemented using hit-and-run, as described in Gilad-Bachrach et al. (2005). For both ALuMA and QBC, we used a fixed number of mixing iterations for hit-and-run, which we set to 1000. We also fixed the number of sampled hypotheses at each iteration of ALuMA to 1000, and used the same set of hypotheses to calculate the majority vote for classification. CAL and QBC examine the examples sequentially, thus the input provided to them was a random ordering of the example pool. The algorithm TK is the first heuristic proposed in Tong and Koller (2002), in which the example chosen at each iteration is the one closest to the max-margin solution of the labeled examples known so far. The graphs below compare the train and the test errors of the different algorithms.

In each of the algorithms, the classification of the training examples is done using the version space defined by the queried labels. The theory for CAL and ERM allows selecting an arbitrary predictor out of the version space. In QBC, the hypothesis should be drawn uniformly at random from the version space. We have found that all the algorithms show a significant improvement in classification error if they classify using

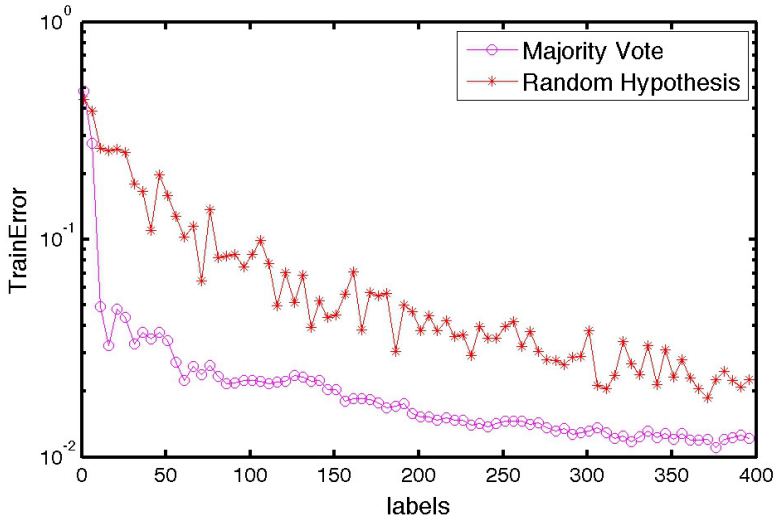


Figure 1: QBC (MNIST 4 vs. 7) - Random hypothesis Vs. Majority vote

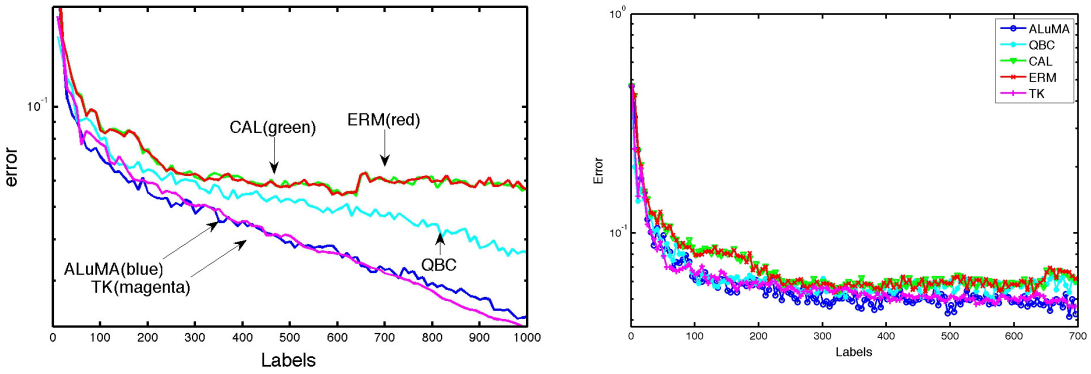


Figure 2: MNIST 3 vs. 5. Train error (left) and test error (right)

the majority vote classification proposed for ALuMA. This observation is demonstrated in Figure 1, which shows the rate of error of QBC (on MNIST data which is described below) using a random hypothesis and a majority vote. Therefore, in all of our experiments below, the results for all the algorithms are based on a majority vote classification.

Our first data set is MNIST.[4] The examples in this data set are gray-scale images of handwritten digits in dimension 784. Each digit has about $6,000$ training examples. We performed binary active learning by pre-selecting pairs of digits. Figure 2 and Figure 3 depict the error as a function of the label budget for two pairs of digits: 3 vs. 5 and 4 vs. 7. It is striking to observe that CAL provides no improvement over passive ERM in the first 1000 examples, while this budget suffices to reach zero training error for ALuMA and TK.

We also tested the algorithms on the PCMAC data set.[5] This is a real-world data set, which represents a two-class categorization of the 20-Newsgroup collection. The examples are web-posts represented using bag-of-words. The original dimension of examples is 7511. We used the Johnson-Lindenstrauss projection to reduce the dimension to 300, which kept the data still separable. We used a training set of 1000 examples.

4. The data set is available at `http://yann.lecun.com/exdb/mnist/`.
5. The data set is available at `http://vikas.sindhwani.org/datasets/lskm/matlab/pcmac.mat`.

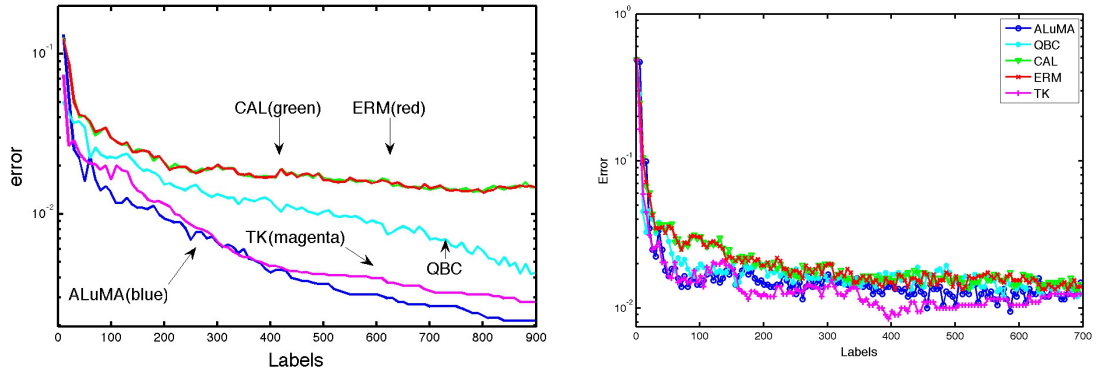


Figure 3: MNIST 4 vs. 7. Train error (left) and test error (right)

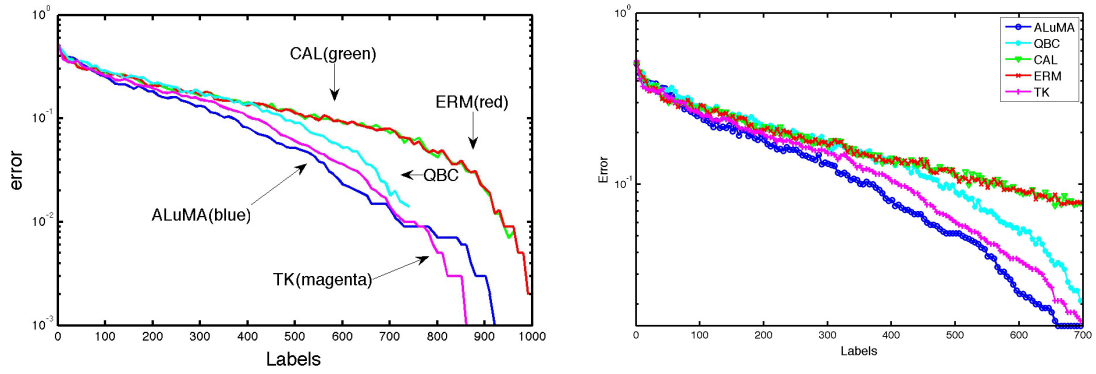


Figure 4: PCMAC. Train error (left) and test error (right)

Figure 4 depicts the results. We were not able to run QBC long enough to use its entire label budget, as it tends to become slower when the training error becomes small.

The following experiments show that ALuMA and TK outperform CAL and QBC even on a data sampled from the uniform distribution on a sphere in $\mathbb{R}^d$. Figure 5 and Figure 6 depict the error as a function of the label budget when learning a random halfspace over the uniform distribution in $\mathbb{R}^{10}$ and $\mathbb{R}^{100}$ respectively.

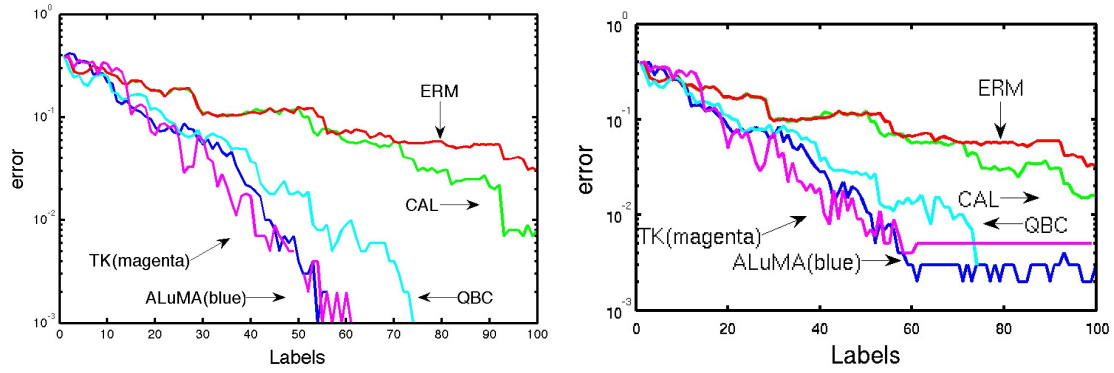


Figure 5: Uniform distribution ($d = 10$). Train error (left) and test error (right)

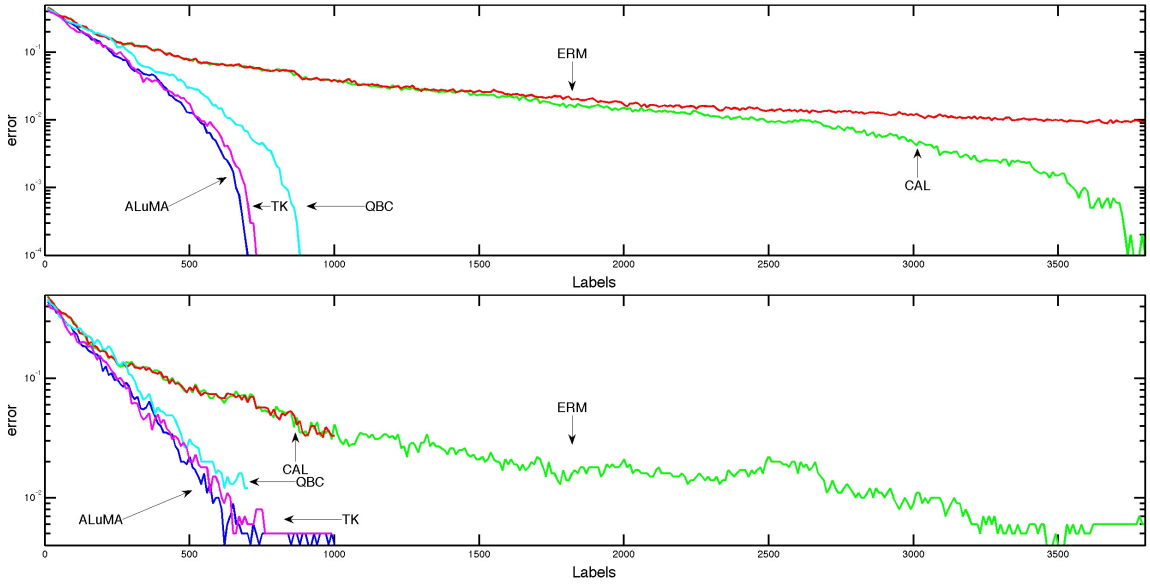


Figure 6: Uniform distribution ($d = 100$). Train error (up) and test error (down)

The difference between the performance of the different algorithms is less marked for $d = 10$ than for $d = 100$, suggesting that the difference grows with the dimension. This result suggests that ALuMA might have a better guarantee than the general relative analysis in the case of the uniform distribution. Achieving such an analysis is an open question which is left for future work.

In the experiments reported so far, TK and ALuMA perform about the same, showing that the TK heuristic is very successful. However, there are cases where TK performs much worse than ALuMA, as the following synthetic experiment demonstrates. In this experiment the pool of examples is taken to be the support of the distribution described in Example 21, with an additional dimension to account for halfspaces with a bias. We also added the negative vertices $-e_i$ to the pool. Similarly to the proof of Theorem 22, it suffices to query the vertices to reach zero error. Table 1 lists the number of iterations required in practice to achieve zero error by each of the algorithms. In this experiment, unlike the rest, ALuMA is not only much better than QBC and CAL, it is also much better than TK, which is worse even than QBC here. This suggests that TK might not have guarantees similar to those of ALuMA, despite the fact that they both attempt to minimize the same objective. The number of queries ALuMA requires is indeed close to the number of vertices.

d	ALuMA	TK	QBC	CAL	ERM
10	**29**	156	50	308	1008
12	**38**	735	113	862	3958
15	**55**	959	150	2401	> 20000

Table 1: Octahedron: number of queries to achieve zero error

To summarize, in all of the experiments above, aggressive algorithms performed better than mellow ones. These results are not fully explained by current theory. The experiments also show that ALuMA and TK have comparable success in practice, but also that there are cases where TK is much worse than ALuMA.

6.3 Non-Separable Data

We now turn to evaluate ALuMA on non-separable data, based on the procedure described in Section 5.2. We compare to IWAL (Beygelzimer et al., 2009), which is a state-of-the-art active learning algorithm for

the agnostic case. We compared ALuMA and IWAL to the passive soft-SVM, which selects random labeled examples from the training set as input.

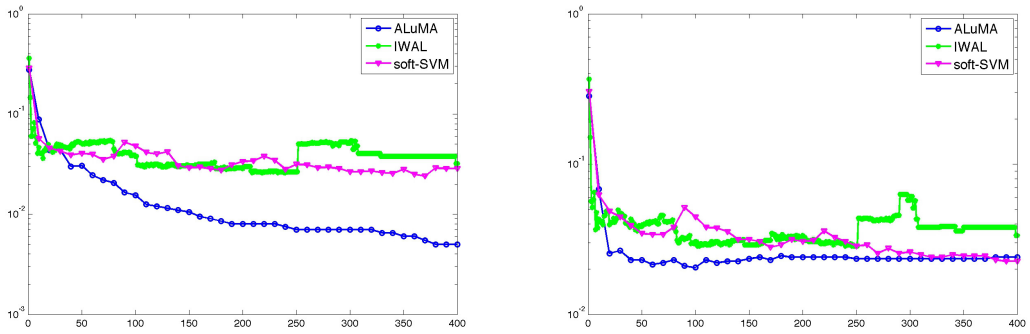


Figure 7: MNIST 4 vs. 7. (non-separable) training error (left) and test error (right)

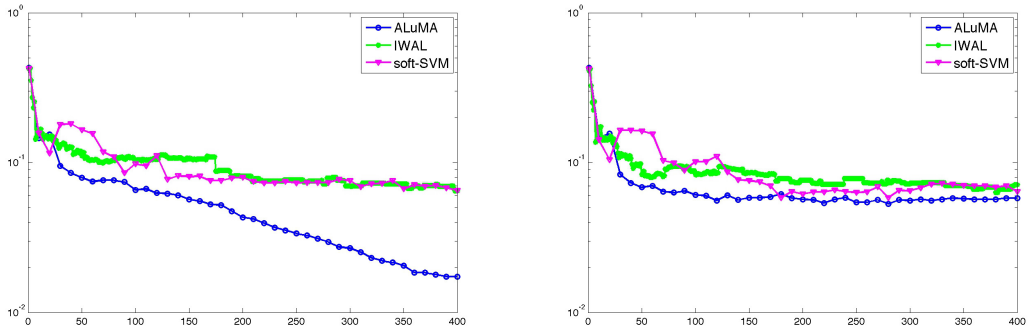


Figure 8: MNIST 3 vs. 5. (non-separable), training error (left) and test error (right)

In our first experiment, we tested the algorithms on the MNIST data, pairs 3 vs. 5 and 4 vs. 7 again, by first reducing the dimension. Following the experimental procedure in Beygelzimer et al. (2009), we projected the 784-dimensional data to a 25-dimensional space using PCA. This renders the two pairs of digits we tested in Section 6.2 non-separable. Using model selection, we set the regularization parameter of soft-SVM to $\lambda = 10^{-3}$ and the maximal norm of the separator in IWAL to $\sqrt{1000}$. For ALuMA, the noise parameter was set to $H = 0.02$ and the dimension after preprocessing was 240. The results are presented in Figures 7 and 8. It can be seen that ALuMA enjoys a faster improvement in error compared to IWAL. This improvement might be attributed to the fact that we assume an upper bound on the hinge-loss in this case, while IWAL must be prepared to handle any amount of label error.

Our second experiment is for the W1A data set.[6] The original data contains a (sparse representation of) more than 2000 train instances and more than $47,000$ test instances in dimension 300. Our preprocessing step used $H = 10^{-2}$ and projected the data to dimension 260. The other parameters were the same as in the previous experiments. The results are shown in Figure 9. It can be seen that in this data set IWAL and ALuMA are comparable, both offering improvement over soft SVM. Unlike MNIST, here ALuMA does not show a consistent improvement over IWAL. We suspect that this is due to the fact that the best achievable error for this data is larger, thus decreasing ALuMA's advantage.

6. The data set is available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

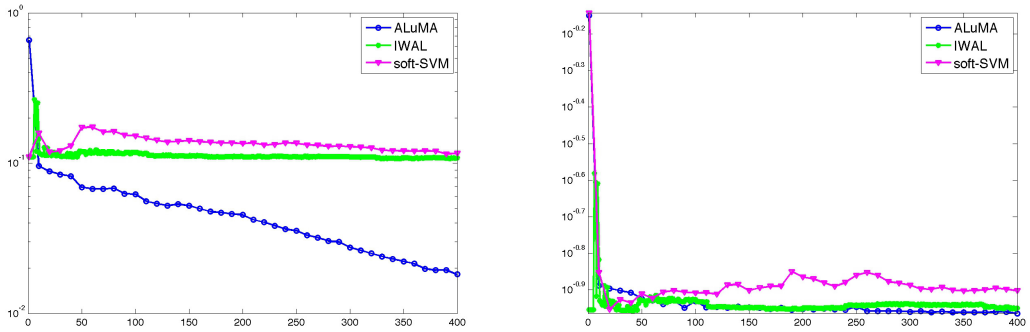


Figure 9: W1A training error (left) and test error (right)

7. Discussion

In this work we have shown that the aggressive approach for active learning can be implemented efficiently and successfully for learning halfspaces. Our theoretical results shed light on the relationship between the margin of the true separator and the number of active queries that the algorithm requires. The experiments show that this approach is practical to implement, and results in improved performance compared to mellow approaches.

Many questions remain open. First, while our analysis guarantees an approximation factor of $O(d\log(m))$, in practice our experiments for the uniform distribution show that in this case the approach performs as well or better than algorithms which are known to achieve almost optimal rates, such as QBC, even in high dimensions. Providing a tight analysis for the label complexity of the aggressive approach for the uniform distribution is thus an interesting open question. Further, while our guarantees only bound the number of queries required to achieve zero error, in practice the algorithm performs well compared to other algorithms even if the goal is only to reach some small non-zero error. Characterizing the behavior of the aggressive approach in this regime is another important open question. Lastly, our work shows that for low-error settings, the aggressive approach can be preferable to the mellow approach. On the other hand, the mellow approach is clearly preferable when error levels are very high. Thus we posit the following open problem for further research: Characterizing the best active learning algorithm one should choose, given a numerical upper bound on the amount of error in the given learning problem.

Appendix A. Proof of Theorem 5

In this section we provide the complete proof of Theorem 5. We will follow Golovin and Krause (2010) and rely on the notion of adaptive sub-modularity.

Denote the product space of partial realizations by $\mathcal{L}_{X,\mathcal{H}}$. Let $f : \mathcal{L}_{X,\mathcal{H}} \to \mathbb{R}_+$ be any utility function from the set of possible partial labelings of X to the non-negative reals. We define the notions of *adaptive monotonicity* and *adaptive submodularity* of a utility function using the following notation: For an element $x \in X$, a subset $Z \subseteq X$ and a hypothesis $h \in \mathcal{H}$, we define the conditional expected marginal benefit of x, conditioned on having observed the partial labeling $h|_Z$, by

$$\Delta(h|_Z, x) = \mathbb{E}_g\left[f(g|_{Z\cup\{x\}}) - f(g|_Z) \,\middle|\, g|_Z = h|_Z\right].$$

Put another way, $\Delta(h|_Z, x)$ is the expected improvement of f if we add to Z the element x, where expectation is over a choice of a hypothesis g taken uniformly at random from the set of hypotheses that agree with h on Z.

Definition 23 (Adaptive Monotonicity) *A utility function $f : \mathcal{L}_{X,\mathcal{H}} \to \mathbb{R}_+$ is* adaptive monotone *if the conditional expected marginal benefit is always non-negative. That is, if for all $h \in \mathcal{H}, Z \subseteq X$ and $x \in X$, $\Delta(h|_Z, x) \geq 0$.*

Definition 24 (Adaptive Submodularity) *A function $f : \mathcal{L}_{X,\mathcal{H}} \to \mathbb{R}_+$ is* adaptive submodular *if the conditional expected marginal benefit of a given item does not increase if the partial labeling is extended. That is, if for all $h \in \mathcal{H}$, for all $Z_1 \subseteq Z_2 \subseteq X$,and for all $x \in X$,*

$$\Delta(h|_{Z_1}, x) \geq \Delta(h|_{Z_2}, x).$$

Any (deterministic) pool-based algorithm is associated with a policy function, which we usually denote by π, which maps each partial realization $h|_S$ to an element x of X, namely, the element x queried by the algorithm after observing $h|_S$. It is natural to consider a greedy algorithm which always selects an item that maximizes the marginal utility. Since it is often computationally hard to choose the element which maximizes the marginal utility, we introduce the notion of an approximately-greedy algorithm, following Golovin and Krause (2010).

Definition 25 (Approximate Greedy) *Let $\alpha \geq 1$. An algorithm which is associated with policy $\pi : \mathcal{L}_{X,\mathcal{H}} \to X$ is* α-approximately greedy *with respect to a utility function f if for every h and for every $Z \subseteq X$*

$$\Delta(h|_Z, \pi(h|_Z)) \geq \frac{1}{\alpha} \max_{x \in X} \Delta(h|_Z, x). \tag{7}$$

If an algorithm $\mathcal{A}$ is 1-approximately greedy *with respect to a utility function f, we simply say that $\mathcal{A}$ is greedy w.r.t. f.*

We denote by $S(\mathcal{A}, h, k)$ the first k pairs of instances along with their labels observed by $\mathcal{A}$, under the assumption that $L \Leftarrow h$. Following this notation, the utility of running $\mathcal{A}$ for k steps under the assumption that $L \Leftarrow h$ is denoted by $f(S(\mathcal{A}, h, k))$. The expected utility of running $\mathcal{A}$ for k steps is defined by

$$f_{\text{avg}}(\mathcal{A}, k) = \mathbb{E}_{h \sim P}[f(S(\mathcal{A}, h, k)].$$

The central theorem of adaptive submodularity, stated below as Theorem 26, links the expected utility of the optimal policy for maximizing f_{avg} with the expected utility of the associated approximately-greedy algorithm.

Theorem 26 (Golovin and Krause (2010)) *Let $f : \mathcal{L}_{X,\mathcal{H}} \to \mathbb{R}_+$ be a utility function, and let $\mathcal{A}$ be a (deterministic) active learning algorithm. If f is adaptive monotone and adaptive submodular, and $\mathcal{A}$ is α-approximately greedy, then for any deterministic algorithm $\mathcal{A}^*$ and for all positive integers t, k,*

$$f_{\text{avg}}(\mathcal{A}, t) \geq (1 - e^{-\frac{t}{\alpha k}}) f_{\text{avg}}(\mathcal{A}^*, k).$$

Let P be a distribution over $\mathcal{H}$. For any algorithm alg, denote by $V_t(\text{alg}, h)$ the version space induced by the first t labels it queries if the true labeling of the pool is consistent with h. Denote the version space reduction of alg after t queries in the case that $L \Leftarrow h$ by

$$f(\text{alg}, t, h) = 1 - P(V_t(\text{alg}, h)). \tag{8}$$

The average version space reduction of alg after t queries is

$$f_{\text{avg}}(\text{alg}, t) = 1 - \mathbb{E}_{h \sim P}[P(V_t(\text{alg}, h))].$$

In the active learning setting, we define the utility function f as in Equation (8) and have the following result:

Lemma 27 (Golovin and Krause (2010)) *The function f defined in Equation (8) is adaptive monotone and adaptive submodular.*

Corollary 28 *Let $X = \{x_1, \ldots, x_m\}$. Let $\mathcal{H}$ be a hypothesis class, and let P be a distribution over $\mathcal{H}$. Suppose that $\mathcal{A}$ is α-approximately greedy with respect to P, and let $\mathcal{A}^*$ be a (deterministic) algorithm that achieves $\mathrm{OPT}_{\max}$, that is $c_{wc}(\mathcal{A}^*) = \mathrm{OPT}_{\max}$. Then, for all positive integers t, k,*

$$f_{\mathrm{avg}}(\mathcal{A}, t) \geq (1 - e^{-\frac{t}{\alpha k}}) f_{\mathrm{avg}}(\mathcal{A}^*, k).$$

The following lemma will allow us to show that the version space of an α-approximately greedy algorithm is relatively pure.

Lemma 29 *Let $\mathcal{A}^*$ be an algorithm that achieves $\mathrm{OPT}_{\max}$. For any $h \in \mathcal{H}$, any active learner $\mathcal{A}$, and any t,*

$$f_{\mathrm{avg}}(\mathcal{A}^*, \mathrm{OPT}_{\max}) - f_{\mathrm{avg}}(\mathcal{A}, t) \geq P(V(h|_X))\left(P(V_t(\mathcal{A}, h)) - P(V(h|_X))\right) .$$

Proof Since $\mathcal{A}^*$ acheives the optimal worst-case cost, the version space induced by the labels that $\mathcal{A}^*$ queries within the first $\mathrm{OPT}_{\max}$ iterations must be exactly the set of hypotheses which are consistent with the true labels of the sample. Therefore, for any $h \in \mathcal{H}$.

$$P(V_{\mathrm{OPT}_{\max}}(\mathcal{A}^*, h)) = P(V(h|_X)).$$

By definition of f_{avg},

$$\begin{aligned} f_{\mathrm{avg}}(\mathcal{A}^*, \mathrm{OPT}_{\max}) - f_{\mathrm{avg}}(\mathcal{A}, t) &= \mathbb{E}_{h\sim P}[P(V_t(\mathcal{A}, h)) - P(V_{\mathrm{OPT}_{\max}}(\mathcal{A}^*, h))] \\ &= \mathbb{E}_{h\sim P}[P(V_t(\mathcal{A}, h)) - P(V(h|_X))]. \end{aligned}$$

Since $S(\mathcal{A}, h, t)$ does not depend on the value of h outside of X, we can sum over the possible labelings of X to have

$$f_{\mathrm{avg}}(\mathcal{A}^*, \mathrm{OPT}_{\max}) - f_{\mathrm{avg}}(\mathcal{A}, t) = \sum_{h|_X : h \in \mathcal{H}} P(V(h|_X))(P(V_t(\mathcal{A}, h)) - P(V(h|_X))).$$

Now, it is easy to see that for any $h \in \mathcal{H}$, $V_t(\mathcal{A}, h) \supseteq V(h|_X)$, thus

$$P(V_t(\mathcal{A}, h)) - P(V(h|_X)) \geq 0.$$

It follows that for any $h \in \mathcal{H}$

$$f_{\mathrm{avg}}(\mathcal{A}^*, \mathrm{OPT}_{\max}) - f_{\mathrm{avg}}(\mathcal{A}, t) \geq P(V(h|_X))(P(V_t(\mathcal{A}, h)) - P(V(h|_X))).$$

■

Combining Corollary 28 and Lemma 29, the following corollary is immediate.

Corollary 30 *For any α-approximate greedy algorithm $\mathcal{A}$,*

$$\forall h \in \mathcal{H}, \quad P(V(h|_X))(P(V_t(\mathcal{A}, h)) - P(V(h|_X))) \;\leq\; e^{-\frac{t}{\alpha \mathrm{OPT}_{\max}}} ,$$

which yields

$$\forall h \in \mathcal{H}, \quad \frac{P(V(h|_X))}{P(V_t(\mathcal{A}, h))} \geq \frac{P(V(h|_X))^2}{e^{-\frac{t}{\alpha \mathrm{OPT}_{\max}}} + P(V(h|_X))^2}. \tag{9}$$

Proof (Of Theorem 5) Let $\mathcal{A}$ be α-approximately greedy algorithm which outputs a β-approximate majority vote. Corollary 30 holds for $\mathcal{A}$. Let h be the target hypothesis. Substituting $T \geq \alpha(2\ln(1/P(h)) + \ln(\frac{\beta}{1-\beta})) \cdot \mathrm{OPT}_{\max}$ into Equation (9) implies that

$$\frac{P(V(h|_X))}{P(V_T(\mathcal{A},h))} \geq \beta \,.$$

The proof now follows from the fact that $\mathcal{A}$ outputs a β-approximate majority vote. ■

Appendix B. Handling Non-Separable Data and Kernel Representations

We now prove Theorem 17 by showing that Alg. 3 satisfies the claims of the theorem. It is clear that Alg. 3 is polynomial as required in item (3). In addition, item (1) holds from the definition of Alg. 3. We have left to prove item (2). We first prove that it holds for the case where the input is represented directly as $X \subseteq \mathbb{R}^d$.

We start by showing that under the assumption of Theorem 17, the set $\{x'_1, \ldots, x'_m\}$, which is generated in step 8, is separated with a bounded margin by the original labels of x_i. Fix $\gamma > 0$ and $w^* \in \mathbb{B}_1^d$. For each $i \in [m]$, define

$$\ell_i = \max(0, \gamma - L(i)\langle w^*, x_i\rangle).$$

Thus, ℓ_i quantifies the margin violation of example x_i by w^*, relative to its true label $L(i)$.

Lemma 31 *If $H \geq \sum_{i=1}^m \ell_i^2$, where H is the input to Alg. 3, then there is a $w \in \mathbb{B}_1^{d+m}$ such that for all $i \in [m]$, $L(i)\langle w, x'_i\rangle \geq \frac{\gamma}{1+\sqrt{H}}$.*

Proof By step 8 in Alg. 3, $x'_i = (a \cdot x_i; \sqrt{1-a^2} \cdot e_i)$, where $a = \sqrt{\frac{1}{1+\sqrt{H}}}$. Define

$$w' = (w^*; \frac{a}{\sqrt{1-a^2}}(L(1)\ell_1, \ldots, L(m)\ell_m)).$$

Then

$$L(i)\langle w', x'_i\rangle = aL(i)\langle w^*, x_i\rangle + a\ell_i \geq a(\gamma - \ell_i) + a\ell_i = a\gamma.$$

Let $w = \frac{w'}{\|w'\|}$. Then $w \in \mathbb{B}_1^{d+m}$, and

$$L(i)\langle w, x'_i\rangle = \frac{L(i)\langle w', x'_i\rangle}{\|w'\|} \geq \frac{a\gamma}{\sqrt{1 + \frac{a^2}{1-a^2}\sum_{i=1}^m \ell_i^2}} = \frac{\gamma}{\sqrt{\frac{1}{a^2} + \frac{1}{1-a^2}\sum_{i=1}^m \ell_i^2}}.$$

Set $a^2 = \frac{1}{1+\sqrt{H}}$, and assume $H \geq \sum_{i=1}^m \ell_i^2$. Then

$$L(i)\langle w, x'_i\rangle \geq \frac{\gamma}{1+\sqrt{H}}.$$

■

The set $\{\bar{x}_1, \ldots, \bar{x}_m\}$ returned by Alg. 3 is a Johnson-Lindenstrauss projection of $\{x'_1, \ldots, x'_m\}$ on $\mathbb{R}^k$. It is known (see, e.g., Balcan et al., 2006b) that if a set of m points is separable with margin η and $k \geq O\left(\frac{\ln(m/\delta)}{\eta^2}\right)$, then with probability $1-\delta$, the projected points are separable with margin $\eta/2$. Setting $\eta = \frac{\gamma}{1+\sqrt{H}}$, it is easy to see that step 12 in Alg. 3 indeed maintains the desired margin. This completes the proof of item (2) of Theorem 17 for the case where the input is $X \subseteq \mathbb{R}^m$.

We now show that if the input is a kernel matrix K, then the decomposition step 3 preserves the separation properties of the input data, thus showing that item (2) holds in this case as well. To show that our decomposition step does not change the properties of the original data, we first use the following lemma, which indicates that separation properties are conserved under different decompositions of the same kernel matrix.

Lemma 32 (Sabato et al. (2010), Lemma 6.3) *Let $K \in \mathbb{R}^{m\times m}$ be a positive definite matrix and let $V \in \mathbb{R}^{m\times n}, U \in \mathbb{R}^{m\times k}$ be matrices such that $K = VV^T = UU^T$. For any vector $w \in \mathbb{R}^n$ there exists a vector $u \in \mathbb{R}^k$ such that $Vw = Uu$ and $\|u\| \leq \|w\|$.*

The next lemma extends the above result, showing that the property holds even if K is not invertible.

Lemma 33 *Let $K \in \mathbb{R}^{m\times m}$ be a positive definite matrix and let $V \in \mathbb{R}^{m\times n}, U \in \mathbb{R}^{m\times k}$ be matrices such that $K = VV^T = UU^T$. For any vector $w \in \mathbb{R}^n$ there exists a vector $u \in \mathbb{R}^k$ such that $Vw = Uu$ and $\|u\| \leq \|w\|$.*

Proof For a matrix A and sets of indexes I, J let $A[I]$ be the sub-matrix of A whose rows are the rows of A with an index in I. Let $A[I,I]$ be the sub-matrix of A whose rows and columns are those that have index I in A.

If K is invertible, the claim holds by Lemma 32. Thus, assume K is singular. Let $I \subseteq [m]$ be a maximal subset such that the matrix $K[I;I]$ is invertible.[7] Then by Lemma 32, $K[I;I] = V[I](V[I])^T = U[I](U[I])^T$, and there exists a vector u such that $V[I]w = U[I]u$, and $\|u\| \leq \|w\|$. We will show that for any $i \notin I$, $V[i]w = U[i]u$ as well.

For any $i \notin I$, $K[I\cup\{i\};I\cup\{i\}]$ is singular. Therefore $V[I\cup\{i\}]$ is singular, while $V[I]$ is not. Thus there is some vector $\lambda \in \mathbb{R}^{|I|}$ such that $V[i]^T = V[I]^T\lambda$. By a similar argument there is some vector $\eta \in \mathbb{R}^{|I|}$ such that $U[i]^T = U[I]^T\eta$. We have $K[I,i] = V[I]V[i]^T = V[I]V[I]^T\lambda = K[I,I]\lambda$. Similarly for U, $K[I,i] = K[I,I]\eta$. Therefore $K[I,I](\lambda-\eta) = 0$. Since $K[I,I]$ is invertible, it follows that $\lambda = \eta$. Therefore, $U[i]u = \eta^T U[I]u = \lambda^T V[I]w = V[i]w$. ■

We now use this lemma to show that the decomposition step does not change the upper bound on the margin loss which is assumed in Theorem 17.

Theorem 34 *Let $\psi_1, \ldots, \psi_m$ be a set of vectors in a Hilbert space S, and let $K \in \mathbb{R}^{m\times m}$ such that for all $i, j \in [m]$, $K_{i,j} = \langle\psi_i, \psi_j\rangle$. suppose there exists a $w \in S$ with $\|w\| \leq 1$ such that*

$$H \geq \sum_{i=1}^{m} \max(0, \gamma - y_i\langle w, \psi_i\rangle)^2. \tag{10}$$

Let $U \in \mathbb{R}^{m\times k}$ such that $K = UU^T$ and let x_i be row i of U. Then there exists a $u \in \mathbb{B}_1^k$ such that

$$H \geq \sum_{i=1}^{m} \max(0, \gamma - y_i\langle u, x_i\rangle)^2. \tag{11}$$

Proof Let $\alpha_1, \ldots, \alpha_n \in S$ be an orthogonal basis for the span of $\psi_1, \ldots, \psi_m$ and w, and let $v_1, \ldots, v_m, v_w \in \mathbb{R}^n$ such that $\sum_{l=1}^n v_i[l]\alpha_l = \psi_i$ and $\sum_{l=1}^n v_w[l]\alpha_l = w$. Let $V \in \mathbb{R}^{m\times n}$ be a matrix such that row i of the matrix is v_i. Then $K = VV^T$, and $Vv_w = r$, where $r[i] = \langle v_w, v_i\rangle = \langle w, \psi_i\rangle$. By Lemma 33, there exists a $u \in \mathbb{R}^k$ such that $Uu = r$. Then we have $\langle u, x_i\rangle = r[i]$. Therefore for all $i \in [m]$, $\langle w, \psi_i\rangle = \langle u, x_i\rangle$, thus Equation (10) implies Equation (11). In addition, $\|u\| \leq \|v_w\| = \|w\| \leq 1$, therefore $u \in \mathbb{B}_1^k$. ■

7. If no such subset exists then K, V, U are all zero and the claim is trivial.

Appendix C. Proof of Theorem 20

Proof [of Theorem 20] Assume that $1/(2\varepsilon)$ is an odd integer and $\varepsilon < 1/8$. Let D_a be the uniform distribution over points on the top circle, defined by

$$S_a = \{a_n \stackrel{\text{def}}{=} (\frac{1}{\sqrt{2}}\cos 2\pi\varepsilon n, \frac{1}{\sqrt{2}}\sin 2\pi\varepsilon n, \frac{1}{\sqrt{2}}) : \ n \in \{0,1,\ldots,1/\varepsilon - 1\}\} \ .$$

Let D_b be the uniform distribution over points on the bottom circle, defined by

$$S_b = \{b_n \stackrel{\text{def}}{=} (\cos 2\pi\varepsilon n, \sin 2\pi\varepsilon n, 0) : \ n \in \{0,1,\ldots,1/\varepsilon - 1\}\} \ .$$

Let $D_{\varepsilon/2}$ be the distribution $(1-\tau)D_a + \tau D_b$, where $\tau = \frac{\varepsilon}{4\log(4/\varepsilon)}$. Note that in order to label $D_{\varepsilon/2}$ correctly with error no more than $\varepsilon/2$, all the labels of points in S_a need to be determined. We prove each of the theorem statements in order. We consider the label complexity with high probability over the choice of unlabeled sample, where high probability is $1-\delta$ for some fixed $\delta \in (0,1/2)$.

First, we prove claim (1). If the unlabeled sample contains only points from S_a, then an active learner has to query all the points in S_a to distinguish between a hypothesis that labels all of S_a positively and one that labels positively all but one point in S_a. Since the probability of the entire set S_b is $o(\varepsilon)$, an i.i.d. sample of size $O(1/\varepsilon)$, will not contain a point from S_b, thus any active learner will require $\Omega(1/\varepsilon)$ labels.

More formally, assume that there exists a constant C and $\varepsilon_0 > 0$ such that if $\varepsilon < \varepsilon_0$, then at most C/ε examples are drawn. Assume from now that $\varepsilon < \varepsilon_0$ and that $\frac{C}{4\log(4/\varepsilon)} \leq 1/2$. Let A be the event that an i.i.d. sample of size $m(\varepsilon) \leq C/\varepsilon$ contains any element from S_b. Then, using the union bound, we obtain

$$\mathbb{P}(A) \leq \frac{C}{\varepsilon}\frac{\varepsilon}{4\log(4/\varepsilon)} \leq 1/2 \leq 1-\delta \ .$$

We now turn to prove claim (2). Assume that the size of the sample is at least $\frac{4\log(4/\varepsilon)\log(1/(\varepsilon\delta))}{\varepsilon^2}$. It is easy to check that with probability at least $1-\delta$, the sample contains all the points in $S_a \cup S_b$. More formally,let $\delta > 0$ be any given confidence parameter. Let B be the event that the sample doesn't contain all the points of D_b and let A the event that the sample doesn't contain all the points of D_a. For $n \in \{0,1,\ldots,1/\varepsilon - 1\}$ let B_n be the event that the sample doesn't contain the element b_n. Then,

$$\mathbb{P}(B_n) = \left(1 - \frac{\varepsilon^2}{4\log(4/\varepsilon)}\right)^{\frac{4\log(4/\varepsilon)\log(2/(\varepsilon\delta))}{\varepsilon^2}} \leq \varepsilon\delta/2 \ .$$

Using the union bound, we obtain that

$$\mathbb{P}(B) \leq \frac{1}{\varepsilon}\mathbb{P}(A_0) \leq \delta/2 \ .$$

Obviously, $\mathbb{P}(A) \leq \mathbb{P}(B)$. Using the union bound, we obtain that with probability at least $1-\delta$, both A and B don't occur.

Given such a sample as a pool, we now show that $\text{OPT}_{\max} = O(\log(1/\varepsilon))$, by describing an active learning algorithm that achieves this label complexity:

1. For all possible separators, the points $b_0 = (1,0,0)$ and $b_{1/2\varepsilon} = (-1,0,0)$ have different labels. The algorithm will first query these initial points, and then apply a binary search to find the boundary between negative and positive labels in S_b. This identifies the labels of all the points in S_b using $O(\log(1/\varepsilon))$ queries.

2. Of the points in S_b, half are labeled positively and half negatively. Moreover, there are n_1, n_2 and $y \in \{-1,1\}$ such that $b_{n_1},\ldots,b_{n_2}$ are all labeled by y, and $n_2 - n_1 + 1 = |S_b|/2 = \frac{1}{2\varepsilon}$ (see illustration in Figure 10). Let $n_3 = \frac{n_2+n_1}{2}$ (this is the middle point with label y). n_3 is an integer because $n_2 - n_1$ is even, thus their sum is also even. Let $n_4 = \mod(n_3 + 1/2\varepsilon, 1/2\varepsilon)$. Query the points a_{n_3} and a_{n_4} for their label.

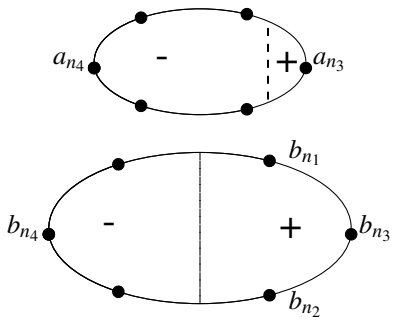


Figure 10: Illustration for the proof of Theorem 20.

3. If a_{n_3} and a_{n_4} each have a different label, apply a binary search starting from these points to find the boundaries between positive and negative labels in S_a, using $O(\log(1/\varepsilon))$ queries. Otherwise, label all the examples in S_a by the label of a_{n_3}.

This algorithm uses $O(\log(1/\varepsilon))$ queries to label the sample. If a_{n_3} and a_{n_4} have different labels, it is clear that the algorithm labels all the examples correctly. We only have left to prove that if they both have the same label, then all the examples in S_a also share that label. Let h^* be the true hypothesis, defined by some homogeneous separator, and assume w.l.o.g that $\{b_n \mid h^*(b_n) = 1\} = \{b_n \in S_b \mid b_n[1] > 0\}$ (note that no point has $b_n[1] = 0$ since $1/2\varepsilon$ is odd). It follows that $n_3 = 0$ and $n_4 = 1/2\varepsilon$, thus $a_{n_3} = (1/\sqrt{2}, 0, 1/\sqrt{2})$ and $a_{n_4} = (-1/\sqrt{2}, 0, 1/\sqrt{2})$ (see illustration in Figure 10). We use the following lemma, whose proof can be found in Appendix D:

Lemma 35 *Assume $1/2\varepsilon$ is odd. If $\{b_n \in S_b \mid h^*(b_n) = 1\} = \{b_n \mid b_n[1] > 0\}$ and $h^*(a_0) = h^*(a_{1/2\varepsilon}) = y$ then $\forall a_n \in S_a, \quad h^*(a_n) = y$.*

If follows that $\mathrm{OPT}_{\max} = O(\log(1/\varepsilon))$.

To bound the label complexity of ALuMA, it suffices to bound from below the minimal margin of possible separators over the given sample. Let h^* be the correct hypothesis. By the same argument as in the proof of Lemma 10, there exists some $w \in \mathbb{R}^3$ that labels the sample identically to h^* and attains its maximal margin on three linearly independent points a, b, c from our sample. Hence, $Aw = \mathbf{1}$ where $A \in \mathbb{R}^{3\times 3}$ is the matrix whose rows are $a, b, c \in S_a \cup S_b$. By Cramer's rule, for every $i \in [3]$

$$w[i] = \frac{\det A_i}{\det A},$$

where A_i is the matrix obtained from A by replacing the i^{th} column with the vector $\mathbf{1}$. Recall that the absolute value of the determinant of A is the volume of the parallelepiped whose sides are a, b and c. Since a, b, c are linearly independent, each of S_a and S_b includes at most two of them. Assume that $a, b \in S_a$ and $c \in S_b$. In this case, the surface area of the basis of this parallelepiped, defined by a and b, is at least $\frac{\sin 2\pi\varepsilon}{\sqrt{2}}$, and the height is $1/\sqrt{2}$. Hence,

$$|\det A| \geq \frac{\sin 2\pi\varepsilon}{2} = \Omega(\varepsilon) \ .$$

The case where two of the points are in S_b leads to an even larger lower bound. Since the elements in each A_i are in $[-1, 1]$, we also have that $|\det A_i| \leq 3! = 6$. Thus, for $i \in [3]$ we obtain that $w_i = O(1/\varepsilon)$. All in all, we get $\|w\|_2 = O(1/\varepsilon)$, and thus $\gamma(h^*) = \Omega(\varepsilon)$. Applying Corollary 9, we obtain that ALuMA classifies all the points correctly using $O(\log(1/\gamma(h^*)) \cdot \mathrm{OPT}_{\max}) = O(\log^2(1/\varepsilon))$ labels.

Finally, we prove claim (3). CAL examines the examples sequentially at a random order, and queries the label of any point whose label is not determined by previous examples. Thus, if the true hypothesis is

all-positive on S_a, and CAL sees all the points in S_a before seeing any point in S_b, it will request $\Omega(1/\varepsilon)$ labels. Hence, it suffices to show that there is a large probability that CAL will indeed examine all of S_a before examining any point from S_b. Let A be the event that the first $\frac{1}{\varepsilon}\log\frac{4}{\varepsilon}$ examples of an i.i.d. sample contain any element from S_b. Then, by the union bound, $\mathbb{P}(A) \leq \frac{1}{\varepsilon}\log(\frac{4}{\varepsilon}) \cdot \frac{\varepsilon}{4\log\frac{4}{\varepsilon}} = 1/4$. Assume now that A does not occur. Let B be the event that the first $\frac{1}{\varepsilon}\log\frac{1}{\varepsilon}$ examples do not contain all the elements in S_a. Then, by the union bound, $\mathbb{P}(B) \leq \frac{1}{\varepsilon}(1-\varepsilon)^{\frac{1}{\varepsilon}\log\frac{4}{\varepsilon}} \leq 1/4$. All in all, with probability at least $1/2$, CAL see all the points in S_a before seeing any point in S_b and thus its label complexity is $\Omega(1/\varepsilon)$. ■

Appendix D. Other Proofs

In this section we provide proofs omitted from the text.

Proof [of Lemma 8] Fix $h \in \mathcal{W}$ and let $V = \{h' \in \mathcal{H} : \forall i, h'(x_i) = h(x_i)\}$. Assume w.l.o.g. that $\|x\| = 1$ for all $x \in X$. Denote for brevity $\gamma = \gamma(h)$. Choose $w \in \mathbb{B}_1^d$ such that $\forall x \in X$, $h(x)\langle w, x\rangle \geq \gamma$. For a given $v \in \mathbb{B}_1^d$, denote by $h_v \in \mathcal{H}$ the mapping $x \mapsto \mathrm{sgn}(\langle v, x\rangle)$. Note that for all $v \in \mathbb{B}_1^d$ such that $\|w - v\| < \gamma$, $h_v \in V$. This is because for all $x \in X$,

$$h(x)\langle v, x\rangle = \langle v - w, h(x)\cdot x\rangle + h(x)\langle w, x\rangle \geq -\|w - v\| \cdot \|h(x)\cdot x\| + \gamma > -\gamma + \gamma = 0,$$

Since $h_v(x) = \mathrm{sgn}(\langle v, x\rangle)$ it follows that $h_v(x) = h(x)$. We conclude that $\{v \mid h_v \in V\} \supseteq \mathbb{B}_1^d \cap B(w, \gamma)$, where $B(z, r)$ denotes the ball of radius r with center at z. Let $u = (1 - \gamma/2)w$. Then for any $z \in B(u, \gamma/2)$, we have $z \in \mathbb{B}_1^d$, since

$$\|z\| = \|z - u + u\| \leq \|z - u\| + \|u\| \leq \gamma/2 + 1 - \gamma/2 = 1.$$

In addition, $z \in B(w, \gamma)$ since

$$\|z - w\| = \|z - u + u - w\| \leq \|z - u\| + \|u - w\| \leq \gamma/2 + \gamma/2 = \gamma.$$

Therefore $B(u, \gamma/2) \subseteq \mathbb{B}_1^d \cap B(w, \gamma)$. We conclude that $\{v \mid h_v \in V\} \supseteq B(u, \gamma/2)$. Thus,

$$P(h) = P(V) \geq \mathrm{Vol}(B(u, \gamma/2))/\mathrm{Vol}(\mathbb{B}_1^d) \geq \left(\frac{\gamma}{2}\right)^d.$$

■

Proof (of Lemma 10) Let us multiply all examples in the pool by $1/c$. Then, all the elements of all examples in the pool are integers. Choose a labeling L which is consistent with some w^*. Consider the optimization problem:

$$\min_w \|w\|^2 \ \text{ s.t. } \ \forall i,\ L(i)\langle w, x_i\rangle \geq 1\ .$$

For simplicity assume that the pool of examples span all of $\mathbb{R}^d$. Then, it is easy to show that if w the solution to the above problem then there exist d linearly independent examples from the pool, denoted w.l.o.g. by $x_1, \ldots, x_d$, such that $L(i)\langle w, x_i\rangle = 1$ for all i. In other words, w is the solution of the linear system $Aw = b$ where the rows of A are $x_1, \ldots, x_d$ and $b = (L(1), \ldots, L(m))^T$.

By Cramer's rule, $w_i = \det(A_i)/\det(A)$, where A_i is obtained by replacing column i of A by the vector b. Since all elements of A are integers and A is invertible, we must have that $|\det(A)| \geq 1$. Therefore, $|w_i| \leq |\det(A_i)|$. Furthermore, by Hadamard's inequality, $|\det(A_i)|$ is upper bounded by the product of the norms of the columns of A_i. Since each element of A_i is upper bounded by $1/c$, we obtain that the norm of each column is at most $\frac{\sqrt{d}}{c}$, hence $|\det(A_i)| \leq (\sqrt{d}/c)^d$. It follows that $\|w\| \leq \sqrt{d}\,(\sqrt{d}/c)^d$. Hence, the margin is

$$\frac{1}{\|w\|\max_i \|x_i\|} \geq \frac{1}{\sqrt{d}\,(\sqrt{d}/c)^d \cdot \sqrt{d}/c} = \frac{1}{\sqrt{d}\,(\sqrt{d}/c)^{d+1}}\ .$$

■

Proof (of Theorem 11) Set $m = \lfloor \ln(1/\gamma) \rfloor$ such that m is a power of 2. Let $x'_0 = (1,0) \in \mathbb{R}^2$. For all $i \in [m-1]$, define $x'_i = (\cos(\pi/2^i), \sin(\pi/2^i))$. Fix $c > 0$, and define $S = \mathbb{B}_1^2 \cap \{-1, -1+c, \ldots, 1-c, 1\}^2$. For each $i \in \{0, 1, \ldots, m-1\}$, let x_i be the nearest neighbor of x'_i in S, that is $x_i = \arg\min_{x \in S} \|x - x'_i\|_2$. It can be easily seen that if $c = \Theta(\gamma)$ then $\forall i$, $\|x_i - x'_i\| = O(\gamma)$.

Consider an exact greedy algorithm that always selects x_0 first (this is possible since on the first round of the algorithm, any query halves the version space). Suppose that the target hypothesis h^* satisfies

$$h^*(x_i) = \begin{cases} -1 & i = 0 \\ 1 & \text{otherwise} \end{cases}$$

By setting a small enough c we get that $\gamma(h^*) = \Omega(\gamma)$.

If c is small enough compared to γ, then after querying x_0 the algorithm will query $x_1, \ldots, x_{m-1}$ in order. In addition, on every round $t < m-1$ the majority vote would lead to the wrong labeling, since only a small fraction of the version space belongs to the correct hypothesis. Thus the algorithm queries all the examples (except perhaps one) before reaching the correct answer. ■

Proof [of Lemma 35] We prove the lemma for the case $h^*(a_{1/2\varepsilon}) = 1$. The case $h^*(a_0) = -1$ can be proved similarly. Let w^* be any hyperplane which is consistent with h^*. Let $n_1 = \frac{1}{4\varepsilon} - \frac{1}{2}$ and let $n_2 = n_1 + 1$. Then

$$b_{n_1} = (\cos(\pi/2 - \pi\varepsilon), \sin(\pi/2 - \pi\varepsilon), 0), \text{ and}$$
$$b_{n_2} = (\cos(\pi/2 + \pi\varepsilon), \sin(\pi/2 + \pi\varepsilon), 0).$$

By the assumption of the lemma, $\langle w^*, b_{n_1} \rangle > 0$ and $\langle w^*, b_{n_2} \rangle < 0$. It follows that $w^*[1] \sin \pi\varepsilon > w^*[2] \cos \pi\varepsilon$ and $-w^*[1] \sin \pi\varepsilon < w^*[2] \cos \pi\varepsilon$. As a consequence, we obtain that $|w^*[2]| < w^*[1] \tan(\pi\varepsilon)$.

Now, choose some $n \in \{0, \ldots, 1/\varepsilon - 1\}$. We show that the corresponding element in S_a is labeled positively. First, from the last inequality, we obtain

$$\begin{aligned} \langle w^*, a_n \rangle &= \frac{1}{\sqrt{2}} \langle w^*, (\cos 2\pi\varepsilon n, \sin 2\pi\varepsilon n, 1) \rangle \\ &\geq \frac{1}{\sqrt{2}} (w_1^* (\cos 2\pi\varepsilon n - \tan(\pi\varepsilon) \sin(2\pi\varepsilon n)) + w_3^*). \end{aligned} \tag{12}$$

We will now show that

$$\forall n \in \{0, 1, \ldots, 1/\varepsilon - 1\}, \ \cos 2\pi\varepsilon n - \tan(\pi\varepsilon) \sin(2\pi\varepsilon n) \geq -1. \tag{13}$$

From symmetry, it suffices to prove this for every $n \in \{0, 1, \ldots, 1/(2\varepsilon) - 1\}$. We divide our range and conclude for each part separately; since $\varepsilon < 1/8$, we have that $\tan \varepsilon\pi < 1$. Then, $\cos\alpha - \tan(\pi\varepsilon) \sin\alpha \geq -1$ in the range $\alpha \in [0, \pi/2]$. For $\alpha \in [\pi/2, \pi - \pi\varepsilon]$, it can be shown that the function $\cos\alpha - \tan(\pi\varepsilon) \sin\alpha$ is monotonically decreasing, thus it suffices to show that the inequality holds for $n = 1/(2\varepsilon) - 1$. Indeed,

$$\begin{aligned} \cos(\pi - 2\varepsilon\pi) - \tan(\varepsilon\pi) \sin(\pi - 2\varepsilon\pi) &= -\cos(2\pi\varepsilon) - 2\sin^2(\pi\varepsilon) \\ &= -\cos^2(\pi\varepsilon) - \sin^2(\pi\varepsilon) \\ &= -1 \ . \end{aligned}$$

Therefore, we obtain from Equation (12) and Equation (13) that

$$\frac{1}{\sqrt{2}} \langle w^*, a_n \rangle \geq \frac{1}{\sqrt{2}} (-w^*[1] + w^*[3]) = \langle w^*, (-1/\sqrt{2}, 0, 1/\sqrt{2}) \rangle = \langle w^*, a_{1/2\varepsilon} \rangle > 0,$$

where the last inequality follows from the assumption that $h^*(a_{1/2\varepsilon}) = 1$. ■

References

D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

E. M. Arkin, H. Meijer, J.S.B. Mitchell, D. Rappaport, and S.S. Skiena. Decision trees for geometric models. In *Proceedings of the Ninth Annual Symposium on Computational Geometry*, pages 369–378. ACM, 1993.

M. F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 65–72. ACM, 2006a.

M. F. Balcan, A. Blum, and S. Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning*, 65(1):79–94, 2006b.

M. F. Balcan, A. Broder, and T. Zhang. Margin based active learning. *Learning Theory*, pages 35–50, 2007.

A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 49–56. ACM, 2009.

J. Bourgain. On lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52(1):46–52, 1985.

G. Brightwell and P. Winkler. Counting linear extensions is #p-complete. In *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing*, STOC '91, pages 175–181, 1991.

D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2): 201–221, 1994.

S. Dasgupta. Analysis of a greedy active learning strategy. *Advances in Neural Information Processing Systems*, 17:337–344, 2005.

S. Dasgupta. Coarse sample complexity bounds for active learning. *Advances in Neural Information Processing Systems*, 18:235, 2006.

S. Dasgupta, A. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. *Learning Theory*, pages 889–905, 2005.

S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. *Advances in Neural Information Processing Systems*, 20:353–360, 2007.

R. El-Yaniv and Y. Wiener. Active learning via perfect selective classification. *The Journal of Machine Learning Research*, 13:255–279, 2012.

Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2):133–168, 1997.

E. Friedman. Active learning for smooth problems. In *Proceedings of the 22nd Conference on Learning Theory*, volume 1, pages 3–2, 2009.

R. Gilad-Bachrach, A. Navot, and N. Tishby. Query by committee made real. *Advances in Neural Information Processing Systems (NIPS)*, 19, 2005.

D. Golovin and A. Krause. Adaptive submodularity: A new approach to active learning and stochastic optimization. In *Proceedings of International Conference on Learning Theory (COLT)*, 2010.

S. Hanneke. A bound on the label complexity of agnostic active learning. In *The 24th Annual International Conference on Machine Learning (ICML)*, 2007.

S. Hanneke. Rates of convergence in active learning. *The Annals of Statistics*, 39(1):333–361, 2011.

J. Håstad. On the size of weights for threshold gates. *SIAM Journal on Discrete Mathematics*, 7:484, 1994.

W. Johnson and J. Lindenstrauss. Extensions of lipschitz mapping into hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

R. Kannan, L. Lovász, and M. Simonovits. Random walks and an $o*(n^5)$ volume algorithm for convex bodies. *Random Structures and Algorithms*, 11(1):1–50, 1997.

L. Lovász. Hit-and-run mixes fast. *Mathematical Programming*, 86(3):443–461, 1999.

J. Matoušek. *Lectures on Discrete Geometry*, volume 212. Springer Verlag, 2002.

A. McCallum and K. Nigam. Employing em in pool-based active learning for text classification. In *Proceedings of ICML-98, 15th International Conference on Machine Learning*, pages 350–358, 1998.

S. Muroga, I. Toda, and S. Takasu. Theory of majority decision elements. *Journal of the Franklin Institute*, 271(5):376–418, 1961.

S. Sabato, N. Srebro, and N. Tishby. Tight sample complexity of large-margin learning. In *Advances in Neural Information Processing Systems 23 (NIPS)*, pages 2038–2046, 2010.

H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 287–294. ACM, 1992.

C. E. Shannon. Probability of error for optimal codes in a gaussian channel. *Bell System Technical Journal*, 38:611–656, 1959.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

Journal of Machine Learning Research 14 (2013) 2617-2640 Submitted 1/13; Revised 5/13; Published 9/13

Keep It Simple And Sparse: Real-Time Action Recognition

Sean Ryan Fanello[*] SEAN.FANELLO@IIT.IT
Ilaria Gori[*] ILARIA.GORI@IIT.IT
Giorgio Metta GIORGIO.METTA@IIT.IT
iCub Facility
Istituto Italiano di Tecnologia
Genova, Via Morego 30, 16163, Italia

Francesca Odone FRANCESCA.ODONE@UNIGE.IT
Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi
Università degli Studi di Genova
Genova, Via Dodecaneso 35, 16146, Italia

Editors: Isabelle Guyon and Vassilis Athitsos

Abstract

Sparsity has been showed to be one of the most important properties for visual recognition purposes. In this paper we show that sparse representation plays a fundamental role in achieving one-shot learning and real-time recognition of actions. We start off from RGBD images, combine motion and appearance cues and extract state-of-the-art features in a computationally efficient way. The proposed method relies on descriptors based on 3D Histograms of Scene Flow (3DHOFs) and Global Histograms of Oriented Gradient (GHOGs); adaptive sparse coding is applied to capture high-level patterns from data. We then propose a simultaneous on-line video segmentation and recognition of actions using linear SVMs. The main contribution of the paper is an effective real-time system for one-shot action modeling and recognition; the paper highlights the effectiveness of sparse coding techniques to represent 3D actions. We obtain very good results on three different data sets: a benchmark data set for one-shot action learning (the ChaLearn Gesture Data Set), an in-house data set acquired by a Kinect sensor including complex actions and gestures differing by small details, and a data set created for human-robot interaction purposes. Finally we demonstrate that our system is effective also in a human-robot interaction setting and propose a memory game, "All Gestures You Can", to be played against a humanoid robot.

Keywords: real-time action recognition, sparse representation, one-shot action learning, human robot interaction

1. Introduction

Action recognition as a general problem is a very fertile research theme due to its strong applicability in several real world domains, ranging from video-surveillance to content-based video retrieval and video classification. This paper refers specifically to action recognition in the context of Human-Machine Interaction (HMI), and therefore it focuses on whole-body actions performed by a human who is standing at a short distance from the sensor.

Imagine a system capable of understanding when to turn the TV on, or when to switch the lights off on the basis of a gesture; the main requirement of such a system is an easy and fast learn-

∗. S.R. Fanello and I. Gori contributed equally to this work.

ing and recognition procedure. Ideally, a single demonstration suffices to teach the system a new gesture. More importantly, gestures are powerful tools, through which languages can be built. In this regard, developing a system able to communicate with deaf people, or to understand paralyzed patients, would represent a great advance, with impact on the quality of life of impaired people. Nowadays these scenarios are likely as a result of the spread of imaging technologies providing real-time depth information at consumer's price (as for example the Kinect (Shotton et al., 2011) by Microsoft); these depth-based sensors are drastically changing the field of action recognition, enabling the achievement of high performance using fast algorithms.

Following this recent trend we propose a *complete system based on RGBD video sequences*, which models actions *from one example only*. Our main goal is to recognize actions in real-time with high accuracy; for this reason we design our system accounting for good performance as well as low computational complexity. The method we propose can be summarized as follows: after segmentation of the moving actor, we extract two types of features from each image, namely, Global Histograms of Oriented Gradient (GHOGs) to model the shape of the silhouette, and 3D Histograms of Flow (3DHOFs) to describe motion information. We then apply a sparse coding stage, which allows us to take care of noise and redundant information and produces a compact and stable representation of the image content. Subsequently, we summarize the action within adjacent frames by building feature vectors that describe the feature evolution over time. Finally, we train a Support Vector Machine (SVM) for each action class.

Our framework can segment and recognize actions accurately and in real-time, even though they are performed in different environments, at different speeds, or combined in sequences of multiple actions. Furthermore, thanks to the simultaneous appearance and motion description complemented by the sparse coding stage, the method provides a one-shot learning procedure. These functions are shown on three different experimental settings: a benchmark data set for one-shot action learning (the ChaLearn Gesture Data Set), an in-house data set acquired by a Kinect sensor including complex actions and gestures differing by small details, and an implementation of the method on a humanoid robot interacting with humans.

In order to demonstrate that our system can be efficiently engaged in real world scenarios, we developed a real-time memory game against a humanoid robot, called "All Gestures You Can" (Gori et al., 2012). Our objective in designing this interaction game is to stress the effectiveness of our gesture recognition system in complex and uncontrolled settings. Nevertheless, our long term goal is to consider more general contexts, which are beyond the game itself, such as rehabilitation and human assistance. Our game may be used also with children with memory impairment, for instance the Attention Deficit/Hyperactivity Disorder (ADHD) (Comoldi et al., 1999). These children cannot memorize items under different conditions, and have low performances during implicit and explicit memory tests (Burden and Mitchell, 2005). Interestingly, Comoldi et al. (1999) shows that when ADHD children were assisted in the use of an appropriate strategy, they performed the memory task as well as controls. The game proposed in this paper could be therefore used to train memory skills to children with attention problems, using the robot as main assistant. The interaction with the robot may increase their motivation to maintain attention and help with the construction of a correct strategy.

The paper is organized as follows: in Section 2 we briefly review the state of the art. In Section 3 sparse representation is presented; Section 4 describes the complete modeling and recognition pipeline. Section 5 validates the approach in different scenarios; Section 6 shows a real application

in the context of Human Robot Interaction (HRI). Finally, Section 7, presents future directions and possible improvements of the current implementation.

2. Related Work

The recent literature is rich of algorithms for gesture, action, and activity recognition—we refer the reader to Aggarwal and Ryoo (2011) and Poppe (2010) for a complete survey of the topic. Even though many theoretically sound, good performing and original algorithms have been proposed, to the best of our knowledge, none of them fulfills at the same time *real-time*, *one-shot learning* and *high accuracy* requirements, although such requirements are all equally important in real world application scenarios.

Gesture recognition algorithms differ in many aspects. A first classification may be done with respect to the overall structure of the adopted framework, that is, how the recognition problem is modeled. In particular, some approaches are based on machine learning techniques, where each action is described as a complex structure; in this class we find methods based on Hidden Markov Models (Malgireddy et al., 2012), Coupled Hidden Semi-Markov models (Natarajan and Nevatia, 2007), action graphs (Li et al., 2010) or Conditional Markov Fields (Chatzis et al., 2013). Other methods are based on matching: the recognition of actions is carried out through a similarity match with all the available data, and the most similar datum dictates the estimated class (Seo and Milanfar, 2012; Mahbub et al., 2011).

The two approaches are different in many ways. Machine learning methods tend to be more robust to intra-class variations, since they distill a model from different instances of the same gesture, while matching methods are more versatile and adapt more easily to one-shot learning, since they do not require a batch training procedure. From the point of view of data representation, the first class of methods usually extracts features from each frame, whereas matching-based methods try to summarize all information extracted from a video in a single feature vector. A recent and prototypical example of machine learning method can be found in Malgireddy et al. (2012), which proposes to extract local features (Histograms of Flow and Histograms of Oriented Gradient) on each frame and apply a bag-of-words step to obtain a global description of the frame. Each action is then modeled as a multi channel Hidden Markov Model (mcHMM). Although the presented algorithm leads to very good classification performance, it requires a computationally expensive offline learning phase that cannot be used in real-time for one-shot learning of new actions. Among the matching-based approaches, Seo and Milanfar (2012) is particularly interesting: the algorithm extract a new type of features, referred to as *3D LSKs*, from space-time regression kernels, particularly appropriate to identify the spatio-temporal geometric structure of the action; it then adopts the Matrix Cosine Similarity measure (Shneider and Borlund, 2007) to perform a robust matching. Another recent method following the trend of matching-based action recognition algorithms is Mahbub et al. (2011); in this work the main features are standard deviation on depth (STD), Motion History Image (MHI) (Bobick and Davis, 2001) and a 2D Fourier Transformation in order to map all information in the frequency domain. This procedure shows some benefits, for instance the invariance to camera shifts. For the matching step, a simple and standard correlation measure is employed. Considering this taxonomy, the work we propose falls within the machine learning approaches, but addresses specifically the problem of one-shot learning. To this end we leverage on the richness of the video signal used as a training example and on a dictionary learning approach to obtain an effective and distinctive representation of the action.

An alternative to classifying gesture recognition algorithms is based on the data representation of gesture models. In this respect there is a predominance of features computed on local areas of single frames (local features), but also holistic features are often used on the whole image or on a region of interest. Among the most known methods, it is worth mentioning the spatio-temporal interesting points (Laptev and Lindeberg, 2003), spatio-temporal Hessian matrices (Willems et al., 2008), Gabor Filters (Bregonzio et al., 2009), Histograms of Flow (Fanello et al., 2010), Histograms of Oriented Gradient (Malgireddy et al., 2012), semi-local features (Wang et al., 2012), combination of multiple features (Laptev et al., 2008), Motion History Image (MHI) (Bobick and Davis, 2001), Space-Time shapes (Gorelick et al., 2007), Self-Similarity Matrices (Efros et al., 2003). Also, due to the recent diffusion of real-time 3D vision technology, 3D features have been recently employed (Gori et al., 2012). For computational reasons as well as the necessity of specific invariance properties, we adopt global descriptors, computed on a region of interest obtained through motion segmentation. We do not rely on a single cue but rather combine motion and appearance similarly to Malgireddy et al. (2012).

The most similar works to this paper are in the field of HMI as for example Lui (2012) and Wu et al. (2012): they both exploit depth information and aim at one-shot learning trying to achieve low computational cost. The first method employs a nonlinear regression framework on manifolds: actions are represented as tensors decomposed via Higher Order Singular Value Decomposition. The underlying geometry of tensor space is used. The second one extracts Extended-MHI as features and uses Maximum Correlation Coefficient (Hirschfeld, 1935) as classifier. Features from RBG and Depth streams are fused via a Multiview Spectral Embedding (MSE). Differently from these works, our approach aims specifically to obtain an accurate real-time recognition from one video example only.

We conclude the section with a reference to some works focusing on continuous action or activity recognition (Ali and Aggarwal, 2001; Green and Guan, 2004; Liao et al., 2006; Alon et al., 2009). In this case training and test videos contain many sequential gestures, therefore the temporal segmentation of videos becomes fundamental. Our work deals with continuous action recognition as well, indeed the proposed framework comprehends a novel and robust temporal segmentation algorithm.

3. Visual Recognition with Sparse Data

One-shot learning is a challenging requirement as the small quantity of training data makes the modeling phase extremely hard. For this reason, in one-shot learning settings a careful choice of the data representation is very important. In this work we rely on sparse coding to obtain a compact descriptor with a good discriminative power even if it is derived from very small data sets.

The main concept behind sparse coding is to approximate an input signal as a linear combination of a few components selected from a dictionary of basic elements, called atoms. We refer to *adaptive sparse coding* when the coding is driven by data. In this case, we require a *dictionary learning* stage, where the dictionary atoms are learnt (Olshausen and Fieldt, 1997; Yang et al., 2009; Wang et al., 2010).

The motivations behind the use of image coding arise from biology: there is evidence that similar signal coding happens in the neurons of the primary visual cortex (V1), which produces sparse and overcomplete activations (Olshausen and Fieldt, 1997). From the computational point of view the objective is to find an overcomplete model of images, unlike methods such as PCA, which

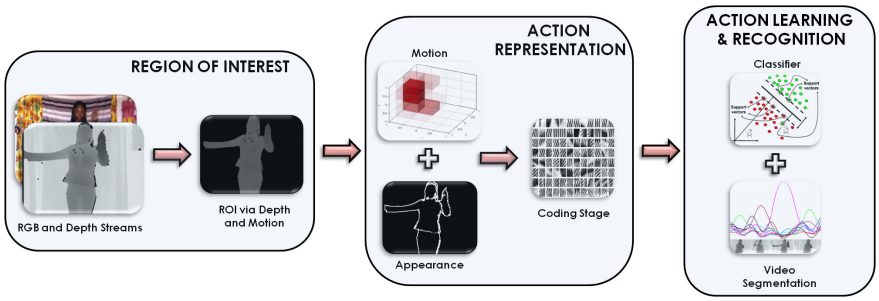


Figure 1: Overview of the recognition system, where video segmentation and classification are performed simultaneously.

aims at finding a number of components that is lower than the data dimensionality. Overcomplete representation techniques have become very popular in applications such as denoising, inpainting, super-resolution, segmentation (Elad and Aharon, 2006; Mairal et al., 2008b,a) and object recognition (Yang et al., 2009). In this work we assess their effectiveness also for gesture recognition. Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m] \in \mathbb{R}^{n \times m}$ be the matrix whose m columns $\mathbf{x}_i \in \mathbb{R}^n$ are the feature vectors. The goal of adaptive sparse coding is to learn a dictionary $\mathbf{D}$ (a $n \times d$ matrix, with d the dictionary size and n the feature vector size) and a code $\mathbf{U}$ (a $d \times m$ matrix) that minimize the reconstruction error:

$$\min_{\mathbf{D},\mathbf{U}} \|\mathbf{X} - \mathbf{D}\mathbf{U}\|_F^2 + \lambda \|\mathbf{U}\|_1, \tag{1}$$

where $\|\cdot\|_F$ is the Frobenius norm. As for the sparsity, it is known that the L_1-norm yields to sparse results while being robust to signals perturbations. Other penalties such as the L_0-norm could be employed, however the problem of finding a solution becomes NP-hard and there is no guarantee that greedy algorithms reach the optimal solution. Notice that fixing $\mathbf{U}$, the above optimization reduces to a least square problem, whilst, given $\mathbf{D}$, it is equivalent to linear regression with the sparsifying norm L_1. The latter problem is referred to as a feature selection problem with a known dictionary (Lee et al., 2007). One of the most efficient algorithms that converges to the optimal solution of the problem in Equation 1 for a fixed $\mathbf{D}$, is the *feature-sign search* algorithm (Lee et al., 2007). This algorithm searches for the sign of the coefficients $\mathbf{U}$; indeed, considering only non-zero elements the problem is reduced to a standard unconstrained quadratic optimization problem (QP), which can be solved analytically. Moreover it performs a refinement of the signs if they are incorrect. For the complete procedure we refer the reader to Lee et al. (2007).

In the context of recognition tasks, it has been proved that a sparsification of the data representation improves the overall classification accuracy (see for instance Guyon and Elisseeff, 2003; Viola and Jones, 2004; Destrero et al., 2009 and references therein). In this case sparse coding is often cast into a *coding-pooling* scheme, which finds its root in the Bag of Words paradigm. In this scheme a *coding operator* is a function $f(\mathbf{x}_i) = \mathbf{u}_i$ that maps $\mathbf{x}_i$ to a new space $\mathbf{u}_i \in \mathbb{R}^k$; when $k > n$ the representation is called overcomplete. The action of coding is followed by a pooling stage, whose purpose is to aggregate multiple local descriptors in a single and global one. Common pooling operators are the max operator, the average operator, or the geometric L_p-norm pooling operator (Feng et al., 2011). More in general, a pooling operator takes the codes located in S regions—for

Figure 2: Region of Interest detection. Left: RGB video frames. Center: depth frames. Right: the detected ROI.

instance cells of the spatial pyramid, as in Yang et al. (2009)—and builds a succinct representation. We define as Y_s the set of locations within the region s. Defining the pooling operator as g, the resultant feature can be rewritten as: $\mathbf{p}_{(s)} = g_{(i \in Y_s)}(\mathbf{u}_{(i)})$. After this stage, a region s of the image is encoded with a single feature vector. The final descriptor of the image is the concatenation of the descriptors $\mathbf{p}_s$ among all the regions. Notice that the effectiveness of pooling is subject to the coding stage. Indeed, if applied on non-coded descriptors, pooling would bring to a drastic loss of information.

4. Action Recognition System

In this section we describe the versatile real-time action recognition system we propose. The system, depicted in Figure 1, consists of three layers, that can be summarized as follows:

- **Region Of Interest detection**: we detect a Region of Interest (ROI), where the human subject is actually performing the action. We use the combination of motion and depth to segment the subject from the background.
- **Action Representation**: each ROI within a frame is mapped into a feature space with a combination of 3D Histogram of Flow (3DHOF) and Global Histogram of Oriented Gradient (GHOG) on the depth map. The resultant 3DHOF+GHOG descriptor is processed via a sparse coding step to compute a compact and meaningful representation of the performed action.
- **Action Learning**: linear SVMs are used on frame buffers. A novel on-line video segmentation algorithm is proposed which allows isolating different actions while recognizing the action sequence.

4.1 Region Of Interest Segmentation

The first step of each action recognition system is to identify correctly where in the image the action is occurring. Most of the algorithms in the literature involve background modeling techniques

(Stauffer and Grimson, 1999), or space-time image filtering in order to extract the interesting spatio-temporal locations of the action (Laptev and Lindeberg, 2003). Other approaches require an *a priori* knowledge of the body pose (Lv and Nevatia, 2007). This task is greatly simplified in our architecture, since in human-machine interaction we can safely assume the human to stand in front of the camera sensors and that there is no other motion in the scene. For each video in the data set, we initially compute the frame differences within consecutive frames in a small buffer, obtaining the set P of pixels that are moving. Relying on this information, we compute the mean depth μ of the pixels belonging to P, which corresponds to the mean depth of the subject within the considered buffer. Thus, for the rest of the video sequence, we select the region of interest as $ROI(t) = \{p_{i,j}(t) : \mu - \varepsilon \leq d(p_{i,j}(t)) \leq \mu + \varepsilon\}$, where $d(p_{i,j}(t))$ is the depth of the pixel $p_{i,j}(t)$ at time t and ε is a tolerance value. In Figure 2 examples of segmentation are shown. We determined empirically that this segmentation procedure achieves better performance with respect to classic thresholding algorithms such as Otsu's method (Otsu, 1979).

4.2 Action Representation

Finding a suitable representation is the most crucial part of any recognition system. Ideally, an image representation should be both *discriminative* and *invariant* to image transformations. A discriminative descriptor should represent features belonging to the same class in a similar way, while it should show low similarity among data belonging to different classes. The invariance property, instead, ensures that image transformations such as rotation, translation, scaling do not affect the final representation. In practice, there is a trade-off between these two properties (Varma and Ray, 2007): for instance, image patches are highly discriminative but not invariant, whereas image histograms are invariant but not discriminative, since different images could be associated to the same representation. When a lot of training data is provided, one could focus on a more discriminative and less invariant descriptor. In our specific case however, where only one training example is provided, invariance is a necessary condition in order to provide discriminant features; this aspect is greatly considered in our method.

From the neuroscience literature it is known that body parts are represented already in the early stages of human development (Mumme, 2001) and that certainly adults have prior knowledge on the body appearance. Many suggests that motion alone can be used to recognize actions (Bisio et al., 2010). In artificial systems this developmental-scale experience is typically not available, although actions can still be represented from two main cues: motion and appearance (Giese and Poggio, 2003). Although many variants of complex features describing human actions have been proposed, many of them imply computationally expensive routines. Differently, we rely on simple features in order to fulfill real-time requirements, and we show that they still have a good discriminative power. In particular we show that a combination of 3D Histograms of Flow (3DHOFs) and Global Histograms of Gradient (GHOGs) models satisfactorily human actions. When a large number of training examples is available, these two features should be able to describe a wide variety of actions, however in one-shot learning scenarios with noisy inputs, they are not sufficient. In this respect, a sparse representation, which keeps only relevant and robust components of the feature vector, greatly simplifies the learning phase making it equally effective.

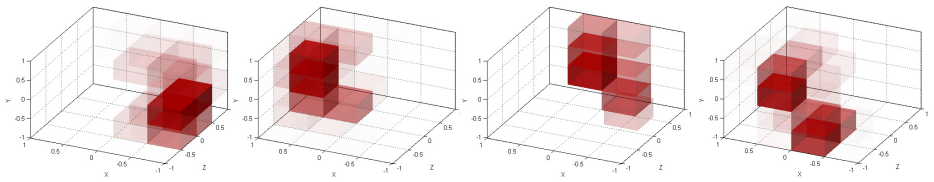


Figure 3: The figure illustrates high level statistics obtained by the proposed scene flow description (3D-HOFs). Starting from the left we show the histogram of the scene flow directions at time t, for a moving hand going on the *Right*, $Left$, *Forward*, *Backward* respectively. Each cuboid represents one bin of the histogram, for visualization purposes we divided the 3D space in $n \times n \times n$ bins with $n = 4$. Filled cuboids represent high density areas.

4.2.1 3D HISTOGRAM OF FLOW

Whereas 2D motion vector estimation has been largely investigated and various fast and effective methods are available today (Papenberg et al., 2006; Horn and Shunk, 1981), the scene flow computation (or 3D motion field estimation) is still an active research field due to the required additional binocular disparity estimation problem. The most promising works are the ones from Wedel et al. (2010), Huguet and Devernay (2007) and Cech et al. (2011); however these algorithms are computationally expensive and may require computation time in the range of 1.5 seconds per frame. This high computational cost is due to the fact that scene flow approaches try to estimate both the 2D motion field and disparity changes. Because of the real-time requirement, we opted for a simpler and faster method that produces a coarser estimation, but is effective for our purposes.

For each frame F_t we compute the 2D optical flow vectors $U(x,y,t)$ and $V(x,y,t)$ for the x and y components with respect to the previous frame F_{t-1}, via the Fanerbäck algorithm (Farnebäck, 2003). Each pixel (x_{t-1}, y_{t-1}) belonging to the ROI of the frame F_{t-1} is reprojected in 3D space $(X_{t-1}, Y_{t-1}, Z_{t-1})$ where the Z_{t-1} coordinate is measured through the depth sensor and X_{t-1}, Y_{t-1} are computed by:

$$\begin{pmatrix} X_{t-1} \\ Y_{t-1} \end{pmatrix} = \begin{pmatrix} \dfrac{(x_{t-1} - x_0) Z_{t-1}}{f} \\ \dfrac{(y_{t-1} - y_0) Z_{t-1}}{f} \end{pmatrix},$$

where f is the focal length and $(x_0, y_0)^T$ is the principal point of the sensor. Similarly, we can reproject the final point (x_t, y_t) of the 2D vector representing the flow, obtaining another 3D vector $(X_t, Y_t, Z_t)^T$. For each pixel of the ROI, we can define the scene flow as the difference of the two 3D vectors in two successive frames F_{t-1} and F_t:

$$\begin{aligned} \mathbf{D} &= (\dot{X}, \dot{Y}, \dot{Z})^T = \\ &= (X_t - X_{t-1}, Y_t - Y_{t-1}, Z_t - Z_{t-1})^T. \end{aligned}$$

Once the 3D flow for each pixel of the ROI at time t has been computed, we normalize it with respect to the $L2$-norm, so that the resulting descriptors $\mathbf{D_1}, \ldots, \mathbf{D_n}$ (n pixels of the ROI) are invariant to the overall speed of the action. In order to extract a compact representation we build a 3D Histogram

of Flow (3DHOF) $\mathbf{z}(t)$ of the 3D motion vectors, where $\mathbf{z}(t) \in \mathbb{R}^{n_1}$ and $\sqrt[3]{n_1}$ is the quantization parameter of the space (i.e., the bin size). In addition we normalize each 3DHOF $\mathbf{z}(t)$ so that $\sum_j z_j(t) = 1$; hence we guarantee that these descriptors are invariant to the subject of interest's scale.

Figure 3 shows that the movements toward different directions reveal to be linearly separable, and the main directions are accurately represented: each cuboid represents one bin of the histogram, and the 3D space is divided in $n \times n \times n$ bins with $n = 4$. It is possible to notice how, in the *Right* direction for example, all the filled bins lay on the semi-space defined by $x < 0$. Similar observations apply all cases.

4.2.2 GLOBAL HISTOGRAM OF ORIENTED GRADIENT

In specific contexts, motion information is not sufficient to discriminate actions, and information on the pose or appearance becomes crucial. One notable example is the American Sign Language (ASL), whose lexicon is based mostly on the shape of the hand. In these cases modeling the shape of a gesture as well as its dynamics is very important. Thus we extend the motion descriptor with a shape feature computed on the depth map. If we assume the subject to be in front of the camera, it is unlikely that the perspective transformation would distort his/her pose, shape or appearance, therefore we can approximately work with invariance to translation and scale. We are interested in characterizing shapes, and the gradient of the depth stream shows the highest responses on the contours, thus studying the orientation of the gradient is a suitable choice. The classical Histograms of Oriented Gradient (HOGs) (Dalal and Triggs, 2005) have been designed for detection purposes and do not show the above-mentioned invariance; indeed dividing the image in cells makes each sub-histogram dependent on the location and the dimension of the object. Furthermore, HOGs exhibit a high spatial complexity, as the classical HOG descriptor belongs to $\mathbb{R}^{(ncells \times nblocks \times n_2)}$. Since we aim at preserving such invariance as well as limiting the computational complexity, we employed a simpler descriptor, the Global Histogram of Oriented Gradient (GHOG). This appearance descriptor produces an overall description of the appearance of the ROI without splitting the image in cells. We compute the histogram of gradient orientations of the pixels on the entire ROI obtained from the depth map to generate another descriptor $\mathbf{h}(t) \in \mathbb{R}^{n_2}$, where n_2 is the number of bins. The scale invariance property is preserved normalizing the descriptor so that $\sum_j h_j(t) = 1$. Computing this descriptor on the depth map is fundamental in order to remove texture information; in fact, in this context, the only visual properties we are interested in are related to shape.

4.2.3 SPARSE CODING

At this stage, each frame F_t is represented by two global descriptors: $\mathbf{z}(t) \in \mathbb{R}^{n_1}$ for the motion component and $\mathbf{h}(t) \in \mathbb{R}^{n_2}$ for the appearance component. Due to the high variability of human actions and to the simplicity of the descriptors, a feature selection stage is needed to catch the relevant information underlying the data and discarding the redundant ones such as background or body parts not involved in the action; to this aim we apply a sparse coding stage to our descriptor.

Given the set of the previously computed 3DHOFs $\mathbf{Z} = [\mathbf{z}(1), \ldots, \mathbf{z}(K)]$, where K is the number of all the frames in the training data, our goal is to learn one motion dictionary $\mathbf{D_M}$ (a $n_1 \times d_1$ matrix, with d_1 the dictionary size and n_1 the motion vector size) and the codes $\mathbf{U_M}$ (a $d_1 \times K$ matrix) that minimize the Equation 1, so that $\mathbf{z}(t) \sim \mathbf{D_M u_M}(t)$. In the same manner, we define the equal optimization problem for a dictionary $\mathbf{D_G}$ (a $n_2 \times d_2$ matrix) and the codes $\mathbf{U_G}$ (a $d_2 \times K$ matrix) for

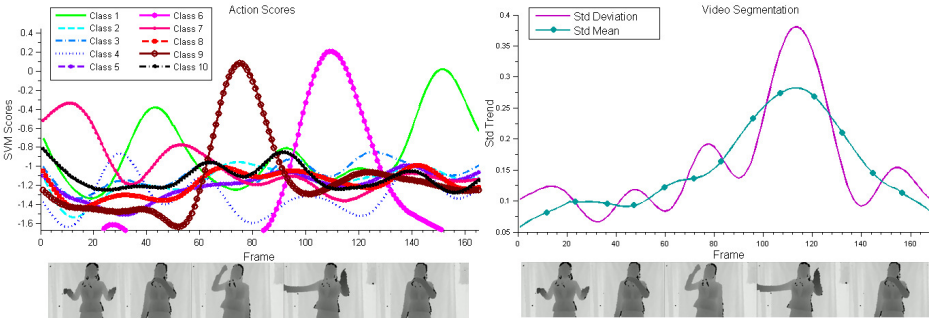


Figure 4: The figure illustrates on the left the SVMs scores (Equation 2) computed in real-time at each time step t over a sequence of 170 frames. On the right the standard deviation of the scores and its mean computed on a sliding window are depicted. The local minima of the standard deviation function are break points that define the end of an action and the beginning of another one. See Section 4.3.2 for details.

the set of GHOGs descriptors $\mathbf{H} = [\mathbf{h}(1), \ldots, \mathbf{h}(K)]$. Therefore, after the Sparse Coding stage, we can describe a frame as a code $\mathbf{u}(i)$, which is the concatenation of the motion and appearance codes: $\mathbf{u}(i) = [\mathbf{u_M}(i), \mathbf{u_G}(i)]$.

Notice that we rely on global features, thus we do not need any pooling operator, which is usually employed to summarize local features into a single one.

4.3 Learning and Recognition

The goal of this phase is to learn a model of a given action from data. Since we are implementing a one-shot action recognition system, the available training data amounts to one training sequence for each action of interest. In order to model the temporal extent of an action we extract sets of sub-sequences from a sequence, each one containing T adjacent frames. In particular, instead of using single frame descriptors (described in Section 4.2), we move to a concatenation of frames: a set of T frames is represented as a sequence $[\mathbf{u}(1), \ldots, \mathbf{u}(T)]$ of codes. This representation allows us to perform simultaneously detection and classification of actions.

The learning algorithm we adopt is the Support Vector Machine (SVM) (Vapnik, 1998). We employ linear SVMs, since they can be implemented with constant complexity during the test phase fulfilling real-time requirements (Fan et al., 2008). Additionally, recent advances in the object recognition field, such as Yang et al. (2009), showed that linear classifiers can effectively solve the classification problem if a preliminary sparse coding stage has previously been applied. Our experiments confirm these findings. Another advantage of linear SVMs is that they can be implemented with a linear complexity in training (Fan et al., 2008); given this property, we can provide a real-time one-shot learning procedure, extremely useful in real applications.

The remainder of the section describes in details the two phases of action learning and action recognition.

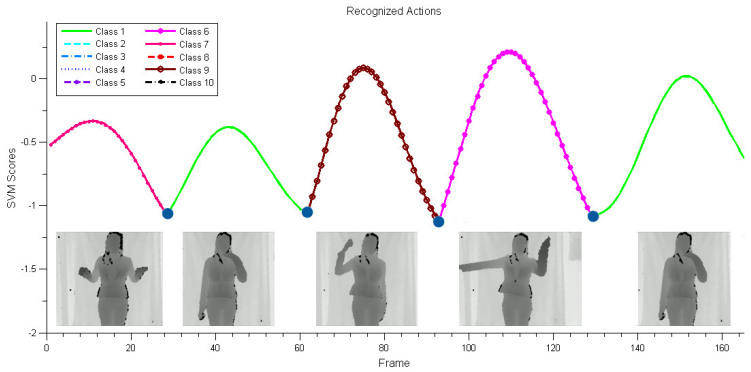


Figure 5: The figure illustrates only the scores of the recognized actions via the method described in Section 4.3.2. Blue dots are the break points computed by the video segmentation algorithm that indicate the end of an action and the beginning of a new one.

4.3.1 ACTION LEARNING

Given a video V_s of t_s frames, containing only one action A_s, we compute a set of descriptors $[\mathbf{u}(1),\ldots,\mathbf{u}(t_s)]$ as described in Section 4.2. Then, action learning is carried out on a set of data that are descriptions of a frame buffer $\mathbf{B}_T(t)$, where T is its length:

$$\mathbf{B}_T(t) = (\mathbf{u}(t-T),\ldots,\mathbf{u}(t-1),\mathbf{u}(t))^T.$$

We use a one-versus-all strategy to train a binary linear SVM for each class A_s, so that at the end of the training phase we obtain a set of N linear SVM classifiers $f_1(\bar{\mathbf{B}}),\ldots,f_N(\bar{\mathbf{B}})$, where N is the number of actions. In particular, in this one-shot learning pipeline, the set of buffers

$$\mathbf{B_s} = [\mathbf{B}_T(t_0),\ldots,\mathbf{B}_T(t_s)]$$

computed from the single video V_s of the class A_s are used as positive examples for the action A_s. All the buffers belonging to A_j with $j \neq s$ are the negative examples. Although we use only one example for each class, we benefit from the chosen representation: indeed, descriptors are computed per frame, therefore one single video of length t_s provides a number of examples equal to $t_s - T$ where T is the buffer size. Given the training data $\{\mathbf{B},\mathbf{y}\}$ where $\mathbf{B}$ is the set of positive and negative examples for the primitive A_s, $y_i = 1$ if the example is positive, $y_i = -1$ otherwise, the goal of SVM is to learn a linear function $(\mathbf{w}^T, b)$ such that a new test vector $\bar{\mathbf{B}}$ is predicted as:

$$y_{pred} = sign(f(\bar{\mathbf{B}})) = sign(\mathbf{w}^{\mathrm{T}}\bar{\mathbf{B}} + b).$$

4.3.2 ON-LINE RECOGNITION: VIDEO SEGMENTATION

Given a test video V, which may contain one or more known actions, the goal is to predict the sequence of the performed actions. The video is analyzed using a sliding window $\mathbf{B}_T(t)$ of size T. We compute the output score $f_i(\mathbf{B}_T(t))$ of the $i = 1,\ldots,N$ SVM machines for each test buffer $\mathbf{B}_T(t)$ and we filter these scores with a low-pass filter W that attenuates noise. Therefore the new score at

time t becomes:

$$H_i(\mathbf{B}_T(t)) = W \star f_i(\mathbf{B}_T(t)) \quad i = 1,\ldots,N, \tag{2}$$

where the $\star$ is the convolution operator. Figure 4 depicts an example of these scores computed in real-time. As long as the scores evolve we need to predict (on-line) when an action ends and another one begins; this is achieved computing the standard deviation $\sigma(H)$ for a fixed t over all the scores H_i^t (Figure 4, right chart). When an action ends we can expect all the SVM output scores to be similar, because no model should be predominant with respect to idle states; this brings to a local minimum in the function $\sigma(H)$. Therefore, each local minimum corresponds to the end of an action and the beginning of a new one. Let n be the number of local minima computed from the standard deviation function; there will be $n+1$ actions, and in particular actions with the highest score before and after each break point will be recognized. We can easily find these minima in real-time: we calculate the mean value of the standard deviation over time using a sliding window. When the standard deviation trend is below the mean, all the SVMs scores predict similar values, hence it is likely that an action has just ended. In Figure 5 the segmented and recognized actions are shown together with their scores.

5. Experiments

In this section we evaluate the performance of our system in three different settings:

- **ChaLearn Gesture Data Set**. The first experiment has been conducted on a publicly available data set, released by ChaLearn (, CGD2011). The main goal of the experiment is to compare our method with other techniques.

- **Kinect Data**. In the second experiment we discuss how to improve the recognition rate using all the functionalities of a real Kinect sensor. Gestures with high level of detail are easily caught by the system.

- **Human-Robot Interaction**. For the last experiment we considered a real HMI scenario: we implement the system on a real robot, the iCub humanoid robot (Metta et al., 2008), showing the applicability of our algorithm also in human-robot interaction settings.

For the computation of the accuracy between a sequence of estimated actions and the ground truth sequence we use the normalized Levenshtein Distance (Levenshtein, 1966), defined as:

$$TeLev = \frac{S+D+I}{M},$$

where each action is treated as a symbol in a sequence, S is the number of substitutions (misclassifications), D the number of deletions (false negatives), I the number of insertions (false positives) and M the length of the ground truth sequence. More specifically, this measure computes the minimum number of modifications that are required to transform a sequence of events in another one. It is widely used in speech recognition contexts, where each symbol represents an event. In action and gesture recognition, when sequences of gestures are to be evaluated, the Levenshtein Distance shows to be a particularly suitable metric, as it allows accounting not only for the single classifier accuracy, but also for the capability of the algorithm to accurately distinguish different gestures in a sequence (Minnen et al., 2006).

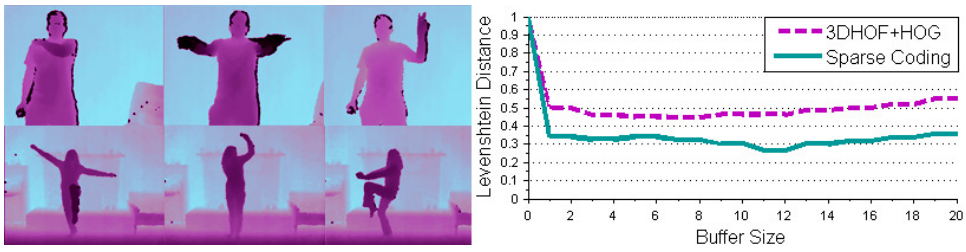


Figure 6: On the left examples of 2 different batches from the ChaLearn Data Set (, CGD2011). On the right the overall Levenshtein Distance computed in 20 batches with respect to the buffer size parameter is depicted for both 3DHOF+GHOG features and descriptors processed with sparse coding.

We empirically choose a quantization parameter for the 3DHOF, n_1 equal to 5, $n_2 = 64$ bins for the GHOG descriptor, and dictionary sizes d_1 and d_2 equal to 256 for both motion and appearance components. This led to a frame descriptor of size 189 for simple descriptors, which increases to 512 after the sparse coding processing. The whole system runs at 25fps on 2.4Ghz Core 2 Duo Processor.

5.1 ChaLearn Gesture Data Set

We firstly assess our method on the ChaLearn data set for the One-Shot Gesture Recognition Challenge (Guyon et al., 2012), see Figure 6. The data set is organized in batches, where each batch includes 100 recorded gestures grouped in sequences of 1 to 5 gestures arbitrarily performed at different speeds. The gestures are drawn from a small vocabulary of 8 to 15 unique gestures called *lexicon*, which is defined within a batch. For each video both RGB and Depth streams are provided, but *only one example* is given for the training phase. In our experiments we do not use information on the body pose of the human. We consider the batches from *devel_01* to *devel_20*; each batch has 47 videos, where L (the lexicon size) videos are for training and the remaining are used as test data.

The main parameter of the system is the buffer size T, however in Figure 6 it is possible to notice that the parameter offers stable performances with a buffer range of $1-20$, so it does not represent a critical variable of our method. Furthermore, high performance for a wide buffer length range imply that our framework is able to handle different speeds implicitly. We compute the Levenshtein Distance as the average over all the batches, which is 25.11% for features processed with sparse coding, whereas simple 3DHOF+GHOG descriptors without sparse coding lead to a performance of 43.32%. Notably, each batch has its own lexicon and some of them are composed of only gestures performed by hand or fingers; in these cases, if the GHOG is computed on the entire ROI, the greatest contribution of the histogram comes from the body shape, whilst finger actions (see Figure 2, bottom row) represent a poor percentage of the final descriptor. If we consider batches where the lexicon is not composed of only hand/fingers gestures, the Levenshtein Distance reduces to 15%.

We compared our method with several approaches. First of all a Template Matching technique, where we used as descriptor the average of all depth frames for each action. The test video is split in

Method	TeLev	TeLen
Sparse Representation **(proposed)**	**25.11%**	**5.02%**
3DHOF + GHOG	43.32%	9.03%
Template Matching	62.56%	15.12%
DTW	49.41%	Manual
Manifold LSR (Lui, 2012)	28.73%	6.24%
MHI (Wu et al., 2012)	30.01%	NA
Extended-MHI (Wu et al., 2012)	26.00%	NA
BoVW (Wu et al., 2012)	72.32%	NA
2D FFT-MHI (Mahbub et al., 2011)	37.46%	NA
TBM+LDA (Malgireddy et al., 2012)	24.09%	NA

Table 1: Levenshtein Distance on the ChaLearn Gesture Data Set. For SVM classification we chose the appropriate buffer size for each batch according to the defined lexicon. TeLev is the Levenshtein Distance, TeLen is the average error (false positives + false negatives) made on the number of gestures (see text).

slices estimated using the average size of actions. In the recognition phase we classify each slice of the video comparing it with all the templates. The overall Levenshtein Distance becomes 62.56%. For the second comparison we employ Dynamic Time Warping (DTW) method (Sakoe and Chiba, 1978) with 3DHOF + GHOG features. We manually divided test videos in order to facilitate the recognition for DTW; nevertheless the global Levenshtein Distance is 49.41%. Finally we report the results presented in some recent works in the field, which exploit techniques based on manifolds (Lui, 2012), Motion History Image (MHI) (Wu et al., 2012), Bag of Visual Words (BoVW) (Wu et al., 2012), 2D FFT-MHI (Mahbub et al., 2011) and Temporal Bayesian Model (TBM) with Latent Dirichlet Allocation (LDA) (Malgireddy et al., 2012).

Table 1 shows that most of the compared approaches are outperformed by our method except for Malgireddy et al. (2012); however the method proposed by Malgireddy et al. (2012) has a training computational complexity of $O(n \times k^2)$ for each action class, where k is the number of HMM states and n the number of examples, while the testing computational complexity for a video frame is $O(k^2)$. Thanks to the sparse representation, we are able to use linear SVMs, which reduce the training complexity with respect to the number of training examples to $O(n \times d)$ for each SVM, where d is the descriptor size. In our case d is a constant value fixed a priori, and does not influence the scalability of the problem. Therefore we may approximate the asymptotic behavior of the SVM in training to $O(n)$. Similarly, in testing the complexity for each SVM is constant with respect to the number of training examples when considering a single frame, and it becomes $O(N)$ for the computation of all the N class scores. This allows us to provide real-time training and testing procedures with the considered lexicons.

Furthermore our on-line video segmentation algorithm shows excellent results with respect to the temporal segmentation used in the compared frameworks; in fact it is worth noting that the proposed algorithm leads to an action detection error rate $TeLen = \frac{FP+FN}{M}$ equal to 5.02%, where FP and FN are false positives and false negatives respectively, and M is the number of all test

gestures. Considering the final results of the ChaLearn Gesture Challenge (Round 1),[1] we placed 9th over 50 teams, but our method also fulfills real-time requirements for the entire pipeline, which was not a requirement of the challenge.

5.1.1 MOTION VS APPEARANCE

In this section we evaluate the contribution of the frame descriptors. In general we notice that the combination of both motion and appearance descriptors leads to the best results when the lexicon is composed of actions where both motion and appearance are equally important. To show this, we considered the 20 development batches from the ChaLearn Gesture Data Set. For this experiment, we used only coded descriptors, since we have already experienced that they obtain higher performance. Using only the motion component, the Levenshtein Distance is equal to 62.89%, whereas a descriptor based only on the appearance leads to an error of 34.15%. The error obtained using only the 3DHOF descriptors was expected, due to the nature of the lexicons chosen: indeed in most gestures the motion component has little significance. Considering instead batch *devel_01*, where motion is an important component in the gesture vocabulary, we have that 3DHOF descriptors lead to a Levenshtein Distance equal to 29.48%, the GHOG descriptors to 21.12% and the combination is equal to 9.11%. Results are consistent with previous findings, but in this specific case the gap between the motion and the appearance components is not critical.

5.1.2 LINEAR VS NON-LINEAR CLASSIFIERS

In this section we compare the performances of linear and non linear SVM for the action recognition task. The main advantage of a linear kernel is the computational time: non-linear SVMs have a worst case training computational complexity per class equal to $O(n^3 \times d)$ against the $O(n \times d)$ of linear SVMs, where n is the number of training examples, and d is the descriptor size. In testing, non linear SVMs show computational complexity of $O(n \times d)$ per frame, since the number of support vectors grows linearly with n. Moreover, non-linear classifiers usually require additional kernel parameter estimation, which especially in one-shot learning scenarios is not trivial. Contrarily, linear SVMs take $O(d)$ per frame. For this experiment we used coded features where both motion and appearance are employed. A non-linear SVM with RBF Kernel has been employed, where the kernel parameter and the SVM regularization term have been chosen empirically after 10 trials on a subset of the batches. The Levenshtein Distance among the 20 batches is 35.11%; this result confirms that linear classifiers are sufficient to obtain good results with low computational cost if an appropriate data representation, as the one offered by sparse coding, is adopted.

5.2 Kinect Data Set

In this section we assess the ability of our method to recognize more complex gestures captured by a Kinect for Xbox 360 sensor. In Section 5.1, we noted that the resolution of the proposed appearance descriptor is quite low and may not be ideal when actions differ by small details, especially on the hands, therefore a localization of the interesting parts to model would be effective. The simplest way to build in this specific information is to resort to a body part tracker; indeed, if a body tracker were available it would have been easy to extract descriptors from different limbs and then concatenate all the features to obtain the final frame representation. An excellent candidate to provide a reliable

1. The leaderboard website is: `https://www.kaggle.com`.

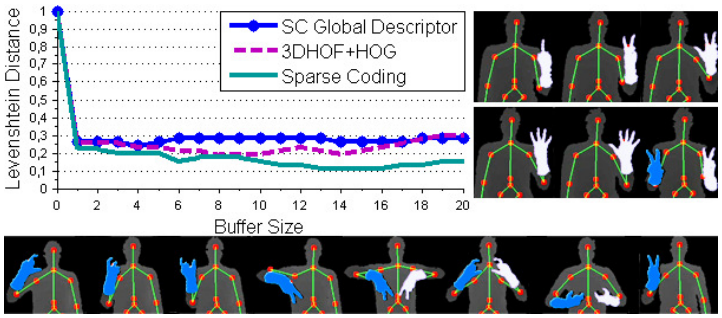


Figure 7: On the right and bottom the two vocabularies used in Section 5.2; these gestures are difficult to model without a proper body tracker, indeed the most contribution for the GHOG comes from the body shape rather than the hand. On the left the Levenshtein Distance.

body tracker is Microsoft Kinect SDK, which implements the method in Shotton et al. (2011). This tool retrieves the 20 principal body joints position and pose of the user's current posture. Given these positions, we assign each 3D point of the ROI to its nearest joint, so that it is possible to correctly isolate the two hands and the body from the rest of the scene (see Figure 7). Then, we slightly modify the approach, computing 3DHOF and GHOG descriptors on three different body parts (left/right hand and whole body shape); the final frame representation becomes the concatenation of all the part descriptors. As for the experiments we have acquired two different sets of data (see Figure 7): in the first one the lexicon is composed of numbers performed with fingers, in the second one we replicate the lexicons *devel_3* of the ChaLearn Gesture Data Set, the one where we obtained the poorest performances. In Figure 7 on the left the overall accuracy is shown; using sparse coding descriptors computed only on the body shape we obtain a Levenshtein Distance around 30%. By concatenating descriptors extracted from the hands the system achieves 10% for features enhanced with sparse coding and 20% for normal descriptors.

We compared our method with two previously mentioned techniques: a Template Matching algorithm and an implementation of the Dynamic Time Warping approach (Sakoe and Chiba, 1978). The resulted Levenshtein Distance is respectively 52.47% and 42.36%.

5.3 Human-Robot Interaction

The action recognition system has been implemented and tested on the iCub, a 53 degrees of freedom humanoid robot developed by the RobotCub Consortium (Metta et al., 2008). The robot is equipped with force sensors and gyroscopes, and it resembles a 3-years old child. It mounts two Dragonfly cameras, providing the basis for 3D vision, thus after an offline camera calibration procedure we can rely on a full stereo vision system; here the depth map is computed following Hirschmuller (2008). In this setting the action recognition system can be used for more general purposes such as Human-Robot-Interaction (HRI) or learning by imitation tasks. In particular our goal is to teach iCub how to perform simple manipulation tasks, such as move/grasp an object. In this sense, we are interested in recognizing actions related to the arm-hand movements of the robot. We define 8 actions, as shown in Figure 8, bottom row, according to the robot manipulation capabilities.

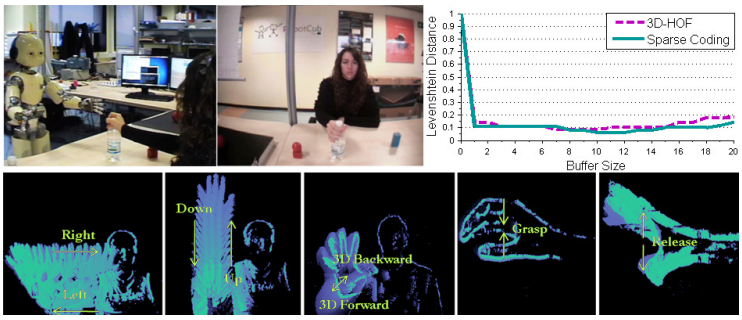


Figure 8: Accuracy for actions sequences (see bottom row). We evaluated the performance on more than 100 actions composed of sequences of 1 to 6 actions.

Each action is modeled using only the motion component (3DHOF), since we want the descriptor to be independent on the particular object shape used.

In Figure 8 we show the accuracy based on the Levenshtein Distance; this measure has been calculated on more than 100 actions composed of sequences of 1 to 6 actions. Notably the error is less than 10%; these good results were expected due to the high discriminative power of the 3DHOFs (Figure 3) on the chosen lexicon, which leads to a linearly separable set.

6. All Gestures You Can: a Real Application

As pointed out in the previous sections, our approach was designed for real applications where real-time requirements need to be fulfilled. We developed and implemented a "game" against a humanoid robot, showing the effectiveness of our system in a real HRI setting: "All Gestures You Can" (Gori et al., 2012), a game aiming at improving memory skills, visual association and concentration. Our game takes inspiration from the classic "Simon" game; nevertheless, since the original version has been often defined as "visually boring", we developed a revisited version, based on gesture recognition, which involves a "less boring" opponent: the iCub (Metta et al., 2008). Both the human and the robot have to take turns and perform the longest possible sequence of gestures by adding one gesture at each turn: one player starts performing a gesture, the opponent has to recognize the gesture, imitate it and add another gesture to the sequence. The game is carried on until one of the two players loses: the human player can lose because of limited memory skills, whereas the robot can lose because the gesture recognition system fails. As described in the previous sections, the system has been designed for one-shot learning; however, Kinect does not provide information about finger configuration, therefore a direct mapping between human fingers and the iCub's ones is not immediate. Thus we set a predefined pool of 8 gestures (see Figure 9, on the left). The typical game setting is shown in Figure 10: the player stays in front of the robot while performing gestures that are recognized with Kinect. Importantly, hand gestures cannot be learned exploiting the Skeleton Data of Kinect: the body tracker detects the position of the hand and it is not enough to discriminate more complicate actions,—for example, see gesture classes 1 and 5 or 2 and 6 in Figure 9.

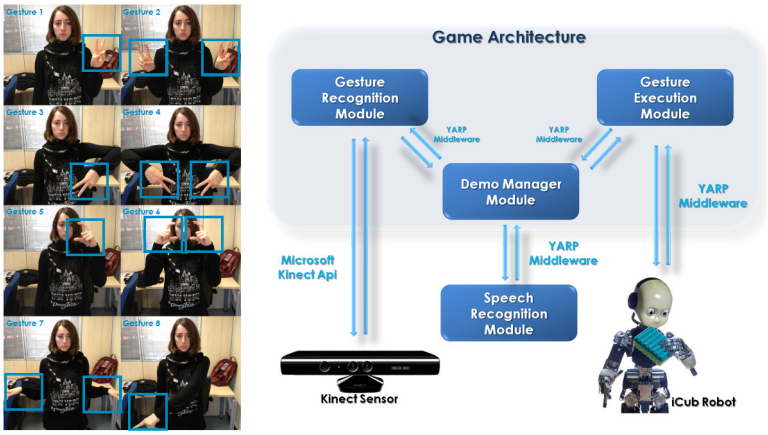


Figure 9: On the left the hand gestures. The vision system has been trained using 8 different actors performing each gesture class for 3 times. On the right the game architecture. There are three main modules that take care of recognizing the action sequence, defining the game rules and making the robot gestures.

The system is simple and modularized as it is organized in three components (see Figure 9) based on the iCub middleware, YARP (Metta et al., 2006), which manages the communication between sensors, processors, and modules. The efficiency of the proposed implementation is assured by its multithreading architecture, which also contributes to real-time performances. The software presented in this section is available in the iCub repository.[2]

The proposed game has been played by more than 30 different players during the ChaLearn Kinect Demonstration Competition at CVPR 2012.[3] Most of them were completely naive without prior knowledge about the gestures. They were asked to play using a lexicon that had been trained specifically for the competition (Figure 9). After 50 matches we had 75% of robot victories. This result indicates that the recognition system is robust also to different players performing variable gestures at various speeds. 15% of the matches have been won by humans and usually they finished during the first 3-4 turns of the game; this always occurred when players performed very different gestures with respect to the trained ones. A few players (10% of matches) succeeded in playing more than 8 turns, and they won due to recognition errors. "All Gestures You Can" ranked 2nd in the ChaLearn Kinect Demonstration Competition.

7. Discussion

This paper presented the design and implementation of a complete action recognition system to be used in real world applications such as HMI. We designed each step of the recognition pipeline to function in real-time while maximizing the overall accuracy. We showed how a sparse action repre-

2. Code available at `https://svn.code.sf.net/p/robotcub/code/trunk/iCub/contrib/src/demoGestureRecognition`.
3. The competition website is `http://gesture.chalearn.org/`
A YouTube video of our game is available at `http://youtu.be/U_JLoe_fT3I`.

Figure 10: The first two turns of a match. Left: the human player performs the first gesture of the sequence. Center: iCub recognized the gesture and imitates it. Right: iCub adds a new random gesture to the sequence.

sentation could be effectively used for one-shot learning of actions in combination with conventional machine learning algorithms (i.e., SVM), even if the latter would normally require a larger set of training data. The comprehensive evaluation of the proposed approach showed that we achieve good trade-off between accuracy and computation time. The main strengths of our learning and recognition pipeline can be summarized as follows:

1. **One-Shot Learning**: one example is sufficient to teach an new action to the system; this is mainly due to the effective per-frame representation.

2. **Sparse Frame Representation**: starting from a simple and computationally inexpensive description that combines global motion (3DHOF) and appearance (GHOG) information over a ROI, subsequently filtered through sparse coding, we obtained a sparse representation at each frame. We showed that these global descriptors are appropriate to model actions of the upper body of a person.

3. **On-line Video Segmentation**: we propose a new, effective, reliable and on-line video segmentation algorithm that achieved a 5% error rate on action detection on a set of 2000 actions grouped in sequences of 1 to 5 gestures. This segmentation procedure works concurrently with the recognition process, thus a sequence of actions is simultaneously segmented and recognized.

4. **Real-time Performances**: the proposed system can be used in real-time applications, as it does require neither a complex features processing nor a computationally expensive training and testing phases. From the computational point of view the proposed approach scales well even for large vocabularies of actions.

5. **Effectiveness in Real Scenarios**: our method achieves good performances in a Human-Robot Interaction setting, where the RGBD images are obtained through binocular vision and disparity estimation. For testing purposes, we proposed a memory game, called "All Gestures You Can", where a person can challenge the iCub robot on action recognition and sequencing. The system ranked 2nd at the Kinect Demonstration Competition.[4]

4. The competition website is `http://gesture.chalearn.org/`.

We stress here the simplicity of the learning and recognition pipeline: each stage is easy to implement and fast to compute. It is shown to be adequate to solve the problem of gesture recognition; we obtained high-quality results while fulfilling real-time requirements. The approach is competitive against many of the state-of-the-art methods for action recognition.

We are currently working on a more precise appearance description at frame level still under the severe constraint of real-time performance; this would enable the use of more complex actions even when the body tracker is not available.

Acknowledgments

This work was supported by the European FP7 ICT projects N. 270490 (EFAA) and N. 270273 (Xperience).

References

J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 2011.

A. Ali and J.K. Aggarwal. Segmentation and recognition of continuous human activity. *IEEE Workshop on Detection and Recognition of Events in Video*, 2001.

J. Alon, V. Athitsos, Y. Quan, and S. Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

A. Bisio, N. Stucchi, M. Jacono, L. Fadiga, and T. Pozzo. Automatic versus voluntary motor imitation: Effect of visual context and stimulus velocity. *PLoS ONE*, 2010.

A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2001.

M. Bregonzio, S. Gong, and T. Xiang. Recognising action as clouds of space-time interest points. *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

M. J. Burden and D. B. Mitchell. Implicit memory development in school-aged children with attention deficit hyperactivity disorder (adhd): Conceptual priming deficit? In *Developmental Neurophysiology*, 2005.

J. Cech, J. Sanchez-Riera, and R. Horaud. Scene flow estimation by growing correspondence seeds. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

ChaLearn Gesture Dataset (CGD2011). `http://gesture.chalearn.org/data`, 2011.

S.P. Chatzis, D. Kosmopoulos, and P. Doliotis. A conditional random field-based model for joint sequence segmentation and classification. In *Pattern Recognition*, 2013.

C. Comoldi, A. Barbieri, C. Gaiani, and S. Zocchi. Strategic memory deficits in attention deficit disorder with hyperactivity participants: The role of executive processes. In *Developmental Neurophysiology*, 1999.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

A. Destrero, C. De Mol, F. Odone, and Verri A. A sparsity-enforcing method for learning face features. *IEEE Transactions on Image Processing*, 18:188–201, 2009.

A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *International Conference on Computer Vision*, 2003.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 2006.

R.-E. Fan, K.-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, 2008.

S. R. Fanello, I. Gori, and F. Pirri. Arm-hand behaviours modelling: from attention to imitation. In *International Symposium on Visual Computing*, 2010.

G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, 2003.

J. Feng, B. Ni, Q. Tian, and S. Yan. Geometric lp-norm feature pooling for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

M. A. Giese and T. Poggio. Neural mechanisms for the recognition of biological movements. *Nature reviews. Neuroscience*, 2003.

L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29, 2007.

I. Gori, S. R. Fanello, F. Odone, and G. Metta. All gestures you can: a memory game against a humanoid robot. *IEEE-RAS International Conference on Humanoid Robots*, 2012.

R.D. Green and L. Guan. Continuous human activity recognition. *Control, Automation, Robotics and Vision Conference*, 2004.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *International Journal of Machine Learning Research*, 3:1157–1182, 2003.

I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hammer, and H. J. E. Balderas. Chalearn gesture challenge: Design and first results. In *Computer Vision and Pattern Recognition Workshops*, 2012.

H. O. Hirschfeld. A connection between correlation and contingency. In *Mathematical Proceedings of the Cambridge Philosophical Society*, 1935.

H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

B. K. P. Horn and B. G. Shunk. Determining optical flow. *Journal of Artificial Intelligence*, 1981.

F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *International Conference on Computer Vision*, 2007.

I. Laptev and T. Lindeberg. Space-time interest points. In *IEEE International Conference on Computer Vision*, 2003.

I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Conference on Neural Information Processing Systems*, 2007.

V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics - Doklady*, 1966.

W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *Computer Vision and Pattern Recognition Workshops*, 2010.

H.-Y.M. Liao, D-Y. Chen, and S.-W Shih. Continuous human action segmentation and recognition using a spatio-temporal probabilistic framework. *IEEE International Symposium on Multimedia*, 2006.

Y. M. Lui. A least squares regression framework on manifolds and its application to gesture recognition. In *Computer Vision and Pattern Recognition Workshops*, 2012.

F. Lv and R. Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

U. Mahbub, H. Imtiaz, T. Roy, S. Rahman, and A. R. Ahad. Action recognition from one example. *Pattern Recognition Letters*, 2011.

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

J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, pages 53–69, 2008b.

M. R. Malgireddy, I. Inwogu, and V. Govindaraju. A temporal Bayesian model for classifying, detecting and localizing activities in video sequences. *Computer Vision and Pattern Recognition Workshops*, 2012.

G. Metta, P. Fitzpatrick, and L. Natale. YARP: Yet Another Robot Platform. *International Journal of Advanced Robotic Systems*, 2006.

G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori. The icub humanoid robot: an open platform for research in embodied cognition. In *Workshop on Performance Metrics for Intelligent Systems*, 2008.

D. Minnen, T. Westeyn, and T. Starner. Performance metrics and evaluation issues for continuous activity recognition. In *Performance Metrics for Intelligent Systems Workshop*, 2006.

D. L. Mumme. Early social cognition: understanding others in the first months of life. *Journal of Infant and Child Development*, 2001.

P. Natarajan and R. Nevatia. Coupled hidden semi markov models for activity recognition. In *Workshop Motion and Video Computing*, 2007.

B. A. Olshausen and D. J. Fieldt. Sparse coding with an overcomplete basis set: a strategy employed by v1. *Vision Research*, 1997.

N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 1979.

N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert. Highly accurate optic flow computation with theoretically justified warping. *International Journal of Computer Vision*, 2006.

R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 2010.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1978.

H. J. Seo and P. Milanfar. A template matching approach of one-shot-learning gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

J. W. Shneider and P. Borlund. Matrix comparison, part 1: Motivation and important issues for measuring the resemblance between proximity measures or ordination results. In *Journal of the American Society for Information Science and Technology*, 2007.

J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

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

M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *IEEE International Conference on Computer Vision*, 2007.

P. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, 2004.

J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3d action recognition with random occupancy patterns. *European Conference on Computer Vision*, 2012.

A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. Stereoscopic scene flow computation for 3d motion understanding. *International Journal of Computer Vision*, 2010.

G. Willems, T. Tuytelaars, and L. Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *European Conference on Computer Vision*, 2008.

D. Wu, F. Zhu, and L. Shao. One shot learning gesture recognition from rgbd images. In *Computer Vision and Pattern Recognition Workshops*, 2012.

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

Journal of Machine Learning Research 14 (2013) 2641-2687 Submitted 3/12; Revised 3/13; Published 9/13

Maximum Volume Clustering: A New Discriminative Clustering Approach*

Gang Niu GANG@SG.CS.TITECH.AC.JP
Department of Computer Science
Tokyo Institute of Technology
2-12-1 O-okayama
Meguro-ku
Tokyo, 152-8552, Japan

Bo Dai BOHR.DAI@GMAIL.COM
College of Computing
Georgia Institute of Technology
801 Atlantic Drive
Atlanta, GA 30332, USA

Lin Shang SHANGLIN@NJU.EDU.CN
State Key Laboratory for Novel Software Technology
Nanjing University
163 Xianlin Avenue
Qixia District
Nanjing 210023, China

Masashi Sugiyama SUGI@CS.TITECH.AC.JP
Department of Computer Science
Tokyo Institute of Technology
2-12-1 O-okayama
Meguro-ku
Tokyo, 152-8552, Japan

Editor: Ulrike von Luxburg

Abstract

The *large volume principle* proposed by Vladimir Vapnik, which advocates that hypotheses lying in an equivalence class with a larger volume are more preferable, is a useful alternative to the *large margin principle*. In this paper, we introduce a new discriminative clustering model based on the large volume principle called *maximum volume clustering* (MVC), and then propose two approximation schemes to solve this MVC model: A soft-label MVC method using *sequential quadratic programming* and a hard-label MVC method using *semi-definite programming*, respectively. The proposed MVC is theoretically advantageous for three reasons. The optimization involved in hard-label MVC is convex, and under mild conditions, the optimization involved in soft-label MVC is akin to a convex one in terms of the resulting clusters. Secondly, the soft-label MVC method pos-

*. A preliminary and shorter version has appeared in Proceedings of 14th International Conference on Artificial Intelligence and Statistics (Niu et al., 2011). The preliminary work was done when GN was studying at Department of Computer Science and Technology, Nanjing University, and BD was studying at Institute of Automation, Chinese Academy of Sciences. A Matlab implementation of maximum volume clustering is available from `http://sugiyama-www.cs.titech.ac.jp/~gang/software.html`.

sesses a *clustering error bound*. Thirdly, MVC includes the optimization problems of a spectral clustering, two relaxed k-means clustering and an information-maximization clustering as *special limit cases* when its regularization parameter goes to infinity. Experiments on several artificial and benchmark data sets demonstrate that the proposed MVC compares favorably with state-of-the-art clustering methods.

Keywords: discriminative clustering, large volume principle, sequential quadratic programming, semi-definite programming, finite sample stability, clustering error bound

1. Introduction

Clustering has been an important topic in machine learning and data mining communities. Over the past decades, a large number of clustering algorithms have been developed. For instance, *k-means clustering* (MacQueen, 1967; Hartigan and Wong, 1979; Girolami, 2002), *spectral clustering* (Shi and Malik, 2000; Meila and Shi, 2001; Ng et al., 2002), *maximum margin clustering* (MMC) (Xu et al., 2005; Xu and Schuurmans, 2005), *dependence-maximization clustering* (Song et al., 2007; Faivishevsky and Goldberger, 2010) and *information-maximization clustering* (Agakov and Barber, 2006; Gomes et al., 2010; Sugiyama et al., 2011). These algorithms have been successfully applied to diverse real-world data sets for exploratory data analysis.

To the best of our knowledge, MMC, which partitions the data samples into two clusters based on the *large margin principle* (LMP) (Vapnik, 1982), is the first clustering approach that is directly connected to the *statistical learning theory* (Vapnik, 1998). For this reason, it has been extensively investigated recently, for example, a generalization (Valizadegan and Jin, 2007) and many approximations for speedup (Zhang et al., 2007; Zhao et al., 2008b,a; Li et al., 2009; Wang et al., 2010).

However, LMP is not the only way to go in statistical learning theory. The *large volume principle* (LVP) was also introduced by Vapnik (1982) for *hyperplanes* and then extended by El-Yaniv et al. (2008) for *soft response vectors*. Roughly speaking, learning methods based on LVP should prefer hypotheses in certain large-volume equivalence classes. See Figure 1 as an illustrative comparison of two principles. Here, C_1, C_2 and C_3 represent three data clouds, and our goal is to choose a better hypothesis from two candidates h_1 and h_2. A hypothesis is a line (e.g., h_1), and an equivalence class is a set of lines which equivalently separate data samples (e.g., H_1). Hence, there are two equivalence classes H_1 and H_2. Given an equivalence class H_1 (or H_2), its margin is measured by the distance between two lines around it and its volume is measured by the area of the region around it in the figure. Though LMP prefers h_1 due to the larger margin of H_1 than H_2, we should choose h_2 when considering LVP since H_2 has a larger volume than H_1.

In this paper, we introduce a novel discriminative clustering approach called *maximum volume clustering* (MVC), which serves as a prototype to partition the data samples into two clusters based on LVP. We motivate our MVC as follows. Given the samples X_n, we construct an X_n-dependent hypothesis space $\mathcal{H}(X_n)$. If $\mathcal{H}(X_n)$ has a measure on it, namely the *power*, then we can talk about the *likelihood* or *confidence* of each equivalence class (Vapnik, 1998). Similarly to the *margin* used in MMC, the notion of *volume* (El-Yaniv et al., 2008) can also be regarded as an estimation of the power. Therefore, the larger the volume is, the more confident we are of the data partition, and we consider the partition lying in the equivalence class with the maximum volume as the best partition.

Similarly to the majority of clustering methods, the optimization problem involved in MVC is combinatorial and thus NP-hard, so we propose two approximation schemes:

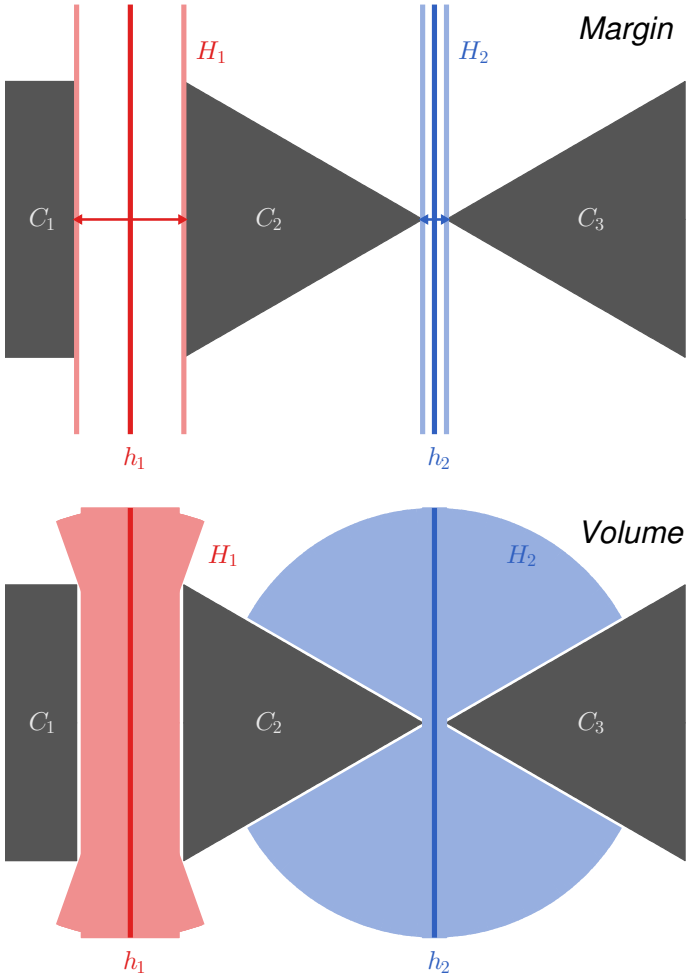


Figure 1: Large margin vs. large volume separation of three data clouds into two clusters. In this figure, the three data clouds are C_1, C_2 and C_3, and the two candidate hypotheses are h_1 and h_2. A hypothesis is a line (e.g., h_1), and an equivalence class is a set of lines which equivalently separate data samples (e.g., H_1). More specifically, we shape H_1 and H_2 by horizontally translating and rotating h_1 and h_2. Then given an equivalence class H_1 (or H_2), its margin is measured by the distance between two lines around it and its volume is measured by the area of the region around it (where we integrate all unit line segments and treat the resulting area as its volume). The large margin principle prefers h_1 and the large volume principle prefers h_2, since they consider different complexity measures.

- A soft-label MVC method that can be solved by *sequential quadratic programming* (Boggs and Tolle, 1995) in $O(n^3)$ time;

- A hard-label MVC method as a *semi-definite programming* problem (De Bie and Cristianini, 2004; Lanckriet et al., 2004) that can be solved in $O(n^{6.5})$ time.

Subsequently, we show that the primal problem of soft-label MVC can be reduced to the optimization problems of *unnormalized spectral clustering* (von Luxburg, 2007), plain and kernel *k-means clustering* after relaxations (Ding and He, 2004), and *squared-loss mutual information based clustering* (Sugiyama et al., 2011), as the regularization parameter of MVC approaches infinity. Hence, MVC might be regarded as a natural extension of many existing clustering methods. Moreover, we establish two theoretical results:

- A theory called *finite sample stability* for analyzing the soft-label MVC method. It suggests that under mild conditions, different locally optimal solutions to soft-label MVC would induce the same data partition, and thus the non-convex optimization of soft-label MVC seems like a convex one;
- A *clustering error bound* for the soft-label MVC method. It upper bounds the distance between the partition returned by soft-label MVC and any partially observed partition based on *transductive Rademacher complexity* (El-Yaniv and Pechyony, 2009).

Experiments on three artificial and fourteen benchmark data sets (i.e., ten IDA benchmarks, USPS, MNIST, 20Newsgroups and Isolet) demonstrate that the proposed MVC approach is promising.

The rest of this paper is organized as follows. First of all, we briefly review the large volume approximation in Section 2. Then, we propose the model and algorithms of MVC in Section 3, and show that they are closely related to several existing clustering methods in Section 4. In Section 5, we present the theory of finite sample stability. In Section 6, we derive the clustering error bound. Next, a comparison with related works is made in Section 7. Experimental results are reported in Section 8. Finally, we give concluding remarks and future prospects in Section 9.

2. Large Volume Approximation

Suppose that we are given a set of objects $X_n = \{x_1, \ldots, x_n\}$, where $x_i \in \mathcal{X}$ for $i = 1, \ldots, n$, and most often but not necessarily, $\mathcal{X} \subset \mathbb{R}^d$ for some natural number d. We will construct a *hypothesis space* $\mathcal{H}(X_n)$ that depends on X_n, such that for any hypothesis $\boldsymbol{h} \in \mathcal{H}(X_n) \subset \mathbb{R}^n$, $[\boldsymbol{h}]_i$ stands for a *soft response* or *confidence-rated label* of x_i, where $[\cdot]_i$ means the i-th component of a vector. We will then pick a *soft response vector* $\boldsymbol{h}^*$ following the large volume principle and partition X_n into two clusters $\{x_i \mid [\boldsymbol{h}^*]_i > 0\}$ and $\{x_i \mid [\boldsymbol{h}^*]_i < 0\}$.[1]

As El-Yaniv et al. (2008), assume that we have a symmetric positive-definite matrix $Q \in \mathbb{R}^{n \times n}$ which contains the pairwise information about X_n. Consider the hypothesis space

$$\mathcal{H}_Q := \{\boldsymbol{h} \mid \boldsymbol{h}^\top Q \boldsymbol{h} \leq 1\},$$

which is geometrically an origin-centered ellipsoid $\mathcal{E}(\mathcal{H}_Q)$ in $\mathbb{R}^n$. The set of sign vectors

$$\{\mathrm{sign}(\boldsymbol{h}) \mid \boldsymbol{h} \in \mathcal{H}_Q\}$$

contains all 2^n possible dichotomies of X_n. In other words, the hypothesis space $\mathcal{H}_Q$ has been partitioned into a finite number of *equivalence classes* $H_1, \ldots, H_{2^n}$, such that for fixed $k \in \{1, 2, \ldots, 2^n\}$,

1. Due to our clustering model that will be defined as optimization (2) in page 2646, $[\boldsymbol{h}^*]_i = 0$ hardly happens in practice, and we simply assume $[\boldsymbol{h}^*]_i \neq 0$ in our problem setting.

all hypotheses in H_k will generate the same dichotomy of X_n. The *power* of an equivalence class H_k is defined as a probability mass

$$\mathcal{P}(H_k) := \int_{H_k} p(\boldsymbol{h}) \mathrm{d}\boldsymbol{h}, \quad k = 1, \ldots, 2^n,$$

where $p(\boldsymbol{h})$ is the underlying probability density of $\boldsymbol{h}$ over $\mathcal{H}_Q$. The hypotheses in H_k with a large power $\mathcal{P}(H_k)$ are preferred according to statistical learning theory (Vapnik, 1998).

When no specific domain knowledge is available (i.e., $p(\boldsymbol{h})$ is unknown), it would be natural to assume the continuous uniform distribution $p(\boldsymbol{h}) = 1/\sum_{k=1}^{2^n} \mathcal{V}(H_k)$, where

$$\mathcal{V}(H_k) := \int_{H_k} \mathrm{d}\boldsymbol{h}, \quad k = 1, \ldots, 2^n,$$

is the *volume* of H_k as well as the geometric volume of the k-th quadrant of $\mathcal{E}(\mathcal{H}_Q)$. Consequently, $\mathcal{P}(H_k)$ is proportional to $\mathcal{V}(H_k)$, and the larger the value of $\mathcal{V}(H_k)$ is, the more confident we are of the data partition $\mathrm{sign}(\boldsymbol{h}^*)$ where $\boldsymbol{h}^*$ is chosen from H_k.

However, it is very hard to accurately compute the geometric volume of a single n-dimensional convex body let alone for all 2^n convex bodies, so we employ an efficient approximation introduced by El-Yaniv et al. (2008) as follows. Let $\lambda_1 \leq \cdots \leq \lambda_n$ be the eigenvalues of Q, and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the associated normalized eigenvectors. Then, $\boldsymbol{v}_i$ and $1/\sqrt{\lambda_i}$ are the direction and length of the i-th principal axis of $\mathcal{E}(\mathcal{H}_Q)$. Note that a small angle from some $\boldsymbol{h} \in H_k$ to $\boldsymbol{v}_i$ with a small/large index i (i.e., a long/short principal axis) implies that $\mathcal{V}(H_k)$ is large/small. Based on this key observation, we define

$$V(\boldsymbol{h}) := \sum_{i=1}^{n} \lambda_i \left(\frac{\boldsymbol{h}^\top \boldsymbol{v}_i}{\|\boldsymbol{h}\|_2} \right)^2 = \frac{\boldsymbol{h}^\top Q \boldsymbol{h}}{\|\boldsymbol{h}\|_2^2}, \tag{1}$$

where $\boldsymbol{h}^\top \boldsymbol{v}_i / \|\boldsymbol{h}\|_2$ means the cosine of the angle between $\boldsymbol{h}$ and $\boldsymbol{v}_i$. We subsequently expect $V(\boldsymbol{h})$ to be small when $\boldsymbol{h}$ lies in a large-volume equivalence class, and conversely to be large when $\boldsymbol{h}$ lies in a small-volume equivalence class.

3. Maximum Volume Clustering

In this section, we define our clustering model and propose two approximation algorithms.

3.1 Basic Formulation

Motivated by Xu et al. (2005), we think of the binary clustering problem from a regularization viewpoint. If we had labels $Y_n = \{y_1, \ldots, y_n\}$ where $y_i \in \{-1, +1\}$, we could find a certain classification method to compute

$$\vartheta(X_n, Y_n) := \min_{\boldsymbol{h} \in \mathcal{H}(X_n, Y_n)} \Delta(Y_n, \boldsymbol{h}) + \gamma W(X_n, \boldsymbol{h}),$$

where $\mathcal{H}(X_n, Y_n)$ is a hypothesis space (which depends upon X_n and Y_n), $\Delta(Y_n, \boldsymbol{h})$ is an overall loss function, $W(X_n, \boldsymbol{h})$ is a regularization function, and $\gamma > 0$ is a regularization parameter. The value of $\vartheta(X_n, Y_n)$ is generally a measure of *classification quality*.

When the labels Y_n are absent, a clustering method tries to minimize $\vartheta(X_n, \boldsymbol{y})$ over all possible assignments $\boldsymbol{y} \in \{-1, +1\}^n$ for given X_n, that is, to solve the problem

$$\boldsymbol{y}^* = \operatorname*{arg\,min}_{\boldsymbol{y} \in \{-1, +1\}^n} \vartheta(X_n, \boldsymbol{y}).$$

Generally speaking, $\vartheta(X_n, \boldsymbol{y}^*)$ can be regarded as a measure of *clustering quality*. The smaller the value of $\vartheta(X_n, \boldsymbol{y}^*)$ is, the more satisfied we are with the resulting data partition $\boldsymbol{y}^*$.

In our discriminative clustering model, we hope to use $V(\boldsymbol{h})$ in Equation (1) as our regularization function. Formally speaking, given the matrix Q, by instantiating $\Delta(\boldsymbol{y}, \boldsymbol{h}) = -2\boldsymbol{h}^\top \boldsymbol{y}$, we define the basic model of *maximum volume clustering* (MVC) as

$$\min_{\boldsymbol{y} \in \{-1,+1\}^n} \min_{\boldsymbol{h} \in \mathcal{H}_Q} -2\boldsymbol{h}^\top \boldsymbol{y} + \gamma \cdot \frac{\boldsymbol{h}^\top Q \boldsymbol{h}}{\|\boldsymbol{h}\|_2^2}, \tag{2}$$

where $\mathcal{H}_Q = \{\boldsymbol{h} \mid \boldsymbol{h}^\top Q \boldsymbol{h} \leq 1\}$ is the hypothesis space mentioned in Section 2, and $\gamma > 0$ is a regularization parameter. Optimization problem (2) is computationally intractable, due to not only the non-convexity of $V(\boldsymbol{h})$ but also the integer feasible region of $\boldsymbol{y}$ which makes (2) combinatorial. In the next two subsections, we will discuss two approximation schemes of (2) in detail.

3.2 Soft-Label Approximation

We now try to optimize $\boldsymbol{h}$ alone by removing $\boldsymbol{y}$. After exchanging the order of the minimizations of $\boldsymbol{y}$ and $\boldsymbol{h}$ in optimization (2), it is easy to see that the optimal $\boldsymbol{y}$ should be $\operatorname{sign}(\boldsymbol{h})$, since the second term is independent of $\boldsymbol{y}$ and the first term is minimized when $\boldsymbol{y} = \operatorname{sign}(\boldsymbol{h})$ for fixed $\boldsymbol{h}$. Therefore, (2) becomes

$$\min_{\boldsymbol{h} \in \mathcal{H}_Q} -2\|\boldsymbol{h}\|_1 + \gamma \cdot \frac{\boldsymbol{h}^\top Q \boldsymbol{h}}{\|\boldsymbol{h}\|_2^2}. \tag{3}$$

Similarly to El-Yaniv et al. (2008), we replace the feasible region $\mathcal{H}_Q$ with $\mathbb{R}^n$, and relax (3) into

$$\min_{\boldsymbol{h} \in \mathbb{R}^n} -2\|\boldsymbol{h}\|_1 + \gamma \boldsymbol{h}^\top Q \boldsymbol{h} \qquad \text{s.t. } \|\boldsymbol{h}\|_2 = 1. \tag{4}$$

Although the optimization is done in $\mathbb{R}^n$, the regularization is done relative to $\mathcal{H}_Q$. Optimization (4) is the primal problem of *soft-label MVC* (MVC-SL).

Optimization (4) is non-convex mainly attributed to the minimization of negative ℓ_1-norm rather than the equality constraint of ℓ_2-norm. In order to solve this optimization, we resort to *sequential quadratic programming* (SQP) (Boggs and Tolle, 1995). The basic idea of SQP is modeling a non-convex problem by a sequence of convex subproblems: At each step, it uses a quadratic model for the objective function and linear models for the constraints. A nonlinear optimization problem with a quadratic objective function and linear constraints is known as *quadratic programming* (QP). An SQP constructs and solves a local QP at each iteration, yielding a step toward the optimum.

More specifically, let us include a class balance constraint $-b \leq \boldsymbol{h}^\top \mathbf{1}_n \leq b$ with a user-specified class balance parameter $b > 0$ to prevent skewed clustering sizes. Denote the objective function of optimization (4) by

$$f(\boldsymbol{h}) := -2\boldsymbol{h}^\top \operatorname{sign}(\boldsymbol{h}) + \gamma \boldsymbol{h}^\top Q \boldsymbol{h},$$

and the auxiliary functions by

$$f_1(\boldsymbol{h}) := \boldsymbol{h}^\top \boldsymbol{h} - 1,$$
$$f_2(\boldsymbol{h}) := \boldsymbol{h}^\top \mathbf{1}_n,$$

where $\mathbf{1}_n$ means the all-one vector in $\mathbb{R}^n$. Subsequently, let λ_1 be the smallest eigenvalue of Q, the corresponding Lagrange function should be[2]

$$L(\boldsymbol{h},\eta,\mu,\nu) = f(\boldsymbol{h}) - \eta f_1(\boldsymbol{h}) - \mu(f_2(\boldsymbol{h}) - b) + \nu(f_2(\boldsymbol{h}) + b),$$

where $\eta < \gamma\lambda_1$ is the Lagrangian multiplier for the constraint $f_1(\boldsymbol{h}) = 0$, and $\mu, \nu \geq 0$ are the Lagrangian multipliers for the constraint $-b \leq f_2(\boldsymbol{h}) \leq b$. Then, given constant $\boldsymbol{h}$ and variable $\boldsymbol{p}$ with a tiny norm, the auxiliary functions can be approximated by

$$\begin{aligned} f_1(\boldsymbol{h}+\boldsymbol{p}) &\approx \boldsymbol{p}^\top \nabla f_1(\boldsymbol{h}) + f_1(\boldsymbol{h}), \\ f_2(\boldsymbol{h}+\boldsymbol{p}) &= \boldsymbol{p}^\top \nabla f_2(\boldsymbol{h}) + f_2(\boldsymbol{h}), \end{aligned}$$

so the constraints are replaced with

$$\begin{aligned} \boldsymbol{p}^\top \nabla f_1(\boldsymbol{h}) + f_1(\boldsymbol{h}) &= 0, \\ -b \leq \boldsymbol{p}^\top \nabla f_2(\boldsymbol{h}) + f_2(\boldsymbol{h}) &\leq b. \end{aligned}$$

Nevertheless, it would be incorrect to adopt the second-order Taylor expansion of $f(\boldsymbol{h}+\boldsymbol{p})$ as our new objective function, since we need to capture the curvature of $f_1(\boldsymbol{h}+\boldsymbol{p})$. The correct way is to use the quadratic model[3]

$$L(\boldsymbol{h}+\boldsymbol{p},\eta) \approx \frac{1}{2}\boldsymbol{p}^\top \nabla^2 L(\boldsymbol{h},\eta)\boldsymbol{p} + \boldsymbol{p}^\top \nabla L(\boldsymbol{h},\eta) + L(\boldsymbol{h},\eta)$$

and form our objective at any fixed $(\boldsymbol{h},\eta)$ into

$$\min_{\boldsymbol{p}\in\mathbb{R}^n} \frac{1}{2}\boldsymbol{p}^\top \nabla^2 L(\boldsymbol{h},\eta)\boldsymbol{p} + \boldsymbol{p}^\top \nabla f(\boldsymbol{h}),$$

according to Boggs and Tolle (1995, p. 9). As a consequence, the subproblem of the t-th iteration is a simple QP at the current estimate $(\boldsymbol{h}_t, \eta_t)$:

$$\begin{aligned} \min_{\boldsymbol{p}_t\in\mathbb{R}^n} \ & \boldsymbol{p}_t^\top(\gamma Q - \eta_t I_n)\boldsymbol{p}_t + 2\boldsymbol{p}_t^\top\big(\gamma Q\boldsymbol{h}_t - \operatorname{sign}(\boldsymbol{h}_t)\big) \\ \text{s.t.} \ & 2\boldsymbol{p}_t^\top\boldsymbol{h}_t + \boldsymbol{h}_t^\top\boldsymbol{h}_t = 1 \\ & -b \leq \boldsymbol{p}_t^\top\mathbf{1}_n + \boldsymbol{h}_t^\top\mathbf{1}_n \leq b, \end{aligned} \tag{5}$$

where I_n is the identity matrix of size n. The new estimate $(\boldsymbol{h}_{t+1}, \eta_{t+1})$ is given by

$$\boldsymbol{h}_{t+1} = \boldsymbol{h}_t + \boldsymbol{p}_t^*, \tag{6}$$

$$\eta_{t+1} = \frac{\boldsymbol{h}_t^\top\big(\gamma Q\boldsymbol{h}_{t+1} - \eta_t\boldsymbol{p}_t^* - \operatorname{sign}(\boldsymbol{h}_t)\big)}{\boldsymbol{h}_t^\top\boldsymbol{h}_t}, \tag{7}$$

2. We will ignore variables μ and ν later, since first-order terms of $L(\boldsymbol{h},\eta,\mu,\nu)$ would disappear in the second-order derivative $\nabla^2 L(\boldsymbol{h},\eta,\mu,\nu)$. The Lagrange function $L(\boldsymbol{h},\eta)$ itself has no constraint on $\boldsymbol{h}$, so we impose $\eta < \gamma\lambda_1$ to make sure that $L(\boldsymbol{h},\eta)$ is bounded from below. Otherwise, the subproblem may be ill-defined.

3. Note that minimizing $-\boldsymbol{h}^\top\boldsymbol{y}$ in optimization (2) or $-\|\boldsymbol{h}\|_1$ in optimization (4) has an effect to push $\boldsymbol{h}$ away from the coordinate axes of $\mathbb{R}^n$. Thus, $[\boldsymbol{h}]_i = 0$ hardly happens in practice and we assume that $\|\boldsymbol{h}\|_1$ is always differentiable.

Algorithm 1 MVC-SL

Input: stopping criterion ε,
symmetric positive-definite matrix Q,
regularization parameter γ,
class balance parameter b
Output: soft response vector $\boldsymbol{h}^*$

Initialize $(\boldsymbol{h}_0, \eta_0)$, recommended but not required, from Equation (9)
$t \leftarrow 0$
repeat
  Obtain $\boldsymbol{p}_t^*$ through optimization (5)
  Update $\boldsymbol{h}_{t+1}$ through Equation (6)
  Update η_{t+1} through Equation (7)
  if $\eta_{t+1} \geq \gamma\lambda_1$ **then break**
  $t \leftarrow t+1$
until $\|\boldsymbol{h}_t - \boldsymbol{h}_{t-1}\|_2 + |\eta_t - \eta_{t-1}| \leq \varepsilon$
return $\boldsymbol{h}^* = \boldsymbol{h}_t$

where $\boldsymbol{p}_t^*$ is the optimal solution to (5). Notice that we cannot obtain η_{t+1} directly from (5) and in fact Equation (7) comes from the best fit in the least-square sense of the following equation

$$\nabla^2 L(\boldsymbol{h}_t, \eta_t)\boldsymbol{p}_t^* + \nabla f(\boldsymbol{h}_t) - \eta_{t+1}\nabla f_1(\boldsymbol{h}_t) = 0. \tag{8}$$

The MVC-SL algorithm based on SQP is summarized in Algorithm 1. In our experiments, we use an initial solution $(\boldsymbol{h}_0, \eta_0)$ defined as

$$\boldsymbol{h}_0 = \frac{1}{\sqrt{n}} \operatorname{sign}\left(\boldsymbol{v}_2 - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top \boldsymbol{v}_2\right), \quad \eta_0 = 0, \tag{9}$$

where $\boldsymbol{v}_2$ is the eigenvector associated with the second smallest eigenvalue of Q. The construction of $\boldsymbol{h}_0$ is explained as follows. The term $(\boldsymbol{v}_2 - \mathbf{1}_n\mathbf{1}_n^\top \boldsymbol{v}_2/n)$ equals $C_n\boldsymbol{v}_2$, where $C_n = I_n - \mathbf{1}_n\mathbf{1}_n^\top/n$ is the centering matrix of size n. It means that we cut $\boldsymbol{v}_2$ at the mean value of its components to form two initial clusters, and normalize the corresponding soft response vector into the unit norm as $\boldsymbol{h}_0$. The asymptotic time complexity of each iteration is at most $O(n^3)$, and the convergence rate of SQP iterations is independent of n (Boggs and Tolle, 1995). Moreover, it takes $O(n^2)$ time to compute $\boldsymbol{h}_0$. Hence, the overall computational complexity of Algorithm 1 is no more than $O(n^3)$.

3.3 Hard-Label Approximation

As opposed to the soft-label approximation, we can also optimize $\boldsymbol{y}$ alone. Let $\boldsymbol{h} = \boldsymbol{\alpha} \circ \boldsymbol{y}$, where $\boldsymbol{\alpha} = |\boldsymbol{h}|$ is a vector of element-wise absolute values, $\boldsymbol{y} = \operatorname{sign}(\boldsymbol{h})$ is a vector of the corresponding signs, and $\circ$ means the element-wise product. We would like to further introduce a hyperparameter C to bound each component of $\boldsymbol{\alpha}$, which might be helpful for dealing with outliers. Subsequently,

the primal problem of *hard-label MVC* (MVC-HL) is written as

$$\begin{aligned} \min_{\boldsymbol{y}\in\{-1,+1\}^n} \min_{\boldsymbol{\alpha}\in\mathbb{R}^n} \; & -2\boldsymbol{\alpha}^\top \mathbf{1}_n + \gamma \boldsymbol{\alpha}^\top \left(Q \circ \boldsymbol{y}\boldsymbol{y}^\top\right) \boldsymbol{\alpha} \\ \text{s.t.} \; & \boldsymbol{\alpha}^\top \boldsymbol{\alpha} = 1 \\ & \mathbf{0}_n \le \boldsymbol{\alpha} \le C\mathbf{1}_n, \end{aligned} \tag{10}$$

where $\mathbf{0}_n$ means the all-zero vector in $\mathbb{R}^n$.

By employing the technique described in Lanckriet et al. (2004), let $M = \boldsymbol{y}\boldsymbol{y}^\top$ and then optimization (10) can be relaxed to

$$\begin{aligned} \min_{M\in\mathbb{R}^{n\times n}} \min_{\eta\in\mathbb{R}} \max_{\boldsymbol{\alpha}\in\mathbb{R}^n} \; & 2\boldsymbol{\alpha}^\top \mathbf{1}_n - \gamma \boldsymbol{\alpha}^\top (Q \circ M)\boldsymbol{\alpha} + \eta(\boldsymbol{\alpha}^\top \boldsymbol{\alpha} - 1) \\ \text{s.t.} \; & M \succeq \mathbf{0} \\ & \operatorname{diag}(M) = \mathbf{1}_n \\ & \mathbf{0}_n \le \boldsymbol{\alpha} \le C\mathbf{1}_n, \end{aligned} \tag{11}$$

where the function $\operatorname{diag}(\cdot)$ forms the diagonal entries of a square matrix into a column vector, and $\succeq \mathbf{0}$ indicates the positive semi-definiteness of a symmetric matrix.[4] The relaxation from (10) to (11) is mainly achieved by replacing $M \in \{-1,+1\}^{n\times n}$ and $\operatorname{rank}(M) = 1$ with $M \in \mathbb{R}^{n\times n}$, $M \succeq \mathbf{0}$ and $\operatorname{diag}(M) = \mathbf{1}_n$. As a result, optimization (11) is a *semi-definite programming* (SDP) provided $(\gamma Q \circ M - \eta I_n) \succeq \mathbf{0}$. Let $\boldsymbol{\nu} \ge \mathbf{0}_n$ and $\boldsymbol{\mu} \ge \mathbf{0}_n$ be the Lagrangian multipliers for $\mathbf{0}_n \le \boldsymbol{\alpha}$ and $\boldsymbol{\alpha} \le C\mathbf{1}_n$, then (11) is equivalent to

$$\begin{aligned} \min_{M,\boldsymbol{\mu},\boldsymbol{\nu},\eta} \max_{\boldsymbol{\alpha}} \; & 2\boldsymbol{\alpha}^\top (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu}) - \boldsymbol{\alpha}^\top (\gamma Q \circ M - \eta I_n)\boldsymbol{\alpha} + 2C\boldsymbol{\mu}^\top \mathbf{1}_n - \eta \\ \text{s.t.} \; & M \succeq \mathbf{0} \\ & \operatorname{diag}(M) = \mathbf{1}_n \\ & \boldsymbol{\mu} \ge \mathbf{0}_n, \boldsymbol{\nu} \ge \mathbf{0}_n. \end{aligned} \tag{12}$$

When considering the variable $\boldsymbol{\alpha}$ in (12), the optimal $\boldsymbol{\alpha}$ should be

$$\boldsymbol{\alpha} = (\gamma Q \circ M - \eta I_n)^\dagger (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu}),$$

where $\dagger$ is the operator of the pseudo inverse, and we can form (12) into

$$\begin{aligned} \min_{M,\boldsymbol{\mu},\boldsymbol{\nu},\eta} \; & (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu})^\top (\gamma Q \circ M - \eta I_n)^\dagger (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu}) + 2C\boldsymbol{\mu}^\top \mathbf{1}_n - \eta \\ \text{s.t.} \; & M \succeq \mathbf{0} \\ & \operatorname{diag}(M) = \mathbf{1}_n \\ & \boldsymbol{\mu} \ge \mathbf{0}_n, \boldsymbol{\nu} \ge \mathbf{0}_n \end{aligned}$$

under an additional condition that $(\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu})$ is orthogonal to the null space of $(\gamma Q \circ M - \eta I_n)$. Eventually, by the *extended Schur complement lemma* (De Bie and Cristianini, 2004), we arrive at

4. We imply by $M \succeq \mathbf{0}$ that M is symmetric and will not explicitly write $M^\top = M$ for convenience.

a standard SDP formulation:

$$\begin{aligned} \min_{M,\boldsymbol{\mu},\boldsymbol{\nu},\eta,t} \quad & t \\ \text{s.t.} \quad & M \succeq \mathbf{0} \\ & \operatorname{diag}(M) = \mathbf{1}_n \\ & \boldsymbol{\mu} \geq \mathbf{0}_n, \boldsymbol{\nu} \geq \mathbf{0}_n \\ & \begin{pmatrix} \gamma Q \circ M - \eta I_n & (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu}) \\ (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu})^\top & t + \eta - 2C\boldsymbol{\mu}^\top \mathbf{1}_n \end{pmatrix} \succeq \mathbf{0}. \end{aligned} \tag{13}$$

The asymptotic time complexity of optimization (13) is $O(n^{6.5})$ if directly solved by any standard SDP solver (De Bie and Cristianini, 2004). It could be reduced to $O(n^{4.5})$ with the *subspace tricks* (De Bie and Cristianini, 2006), which essentially make use of the spectral properties of Q to control the trade-off between the computational cost and the accuracy.

After we obtain M^*, $\boldsymbol{y}^*$ could be recovered from the rank one approximation of M^* by either *thresholding* (De Bie and Cristianini, 2004) or *randomized rounding* (Raghavan and Thompson, 1985; De Bie and Cristianini, 2006). In our experiments, we use the former technique: The eigenvector $\boldsymbol{v}^*$ associated with the largest eigenvalue of M^* is extracted, and then $\boldsymbol{y}^*$ is recovered as

$$\boldsymbol{y}^* = \operatorname{sign}\left(\boldsymbol{v}^* - \frac{1}{n}\mathbf{1}_n \mathbf{1}_n^\top \boldsymbol{v}^*\right),$$

where the threshold is the center of $\boldsymbol{v}^*$ (cf., the construction of $\boldsymbol{h}_0$ in MVC-SL).

4. Generality

MVC is a general framework and closely related to several existing clustering methods. The primal problem of MVC-SL can in fact be reduced to the optimization problems of *unnormalized spectral clustering* (USC) (von Luxburg, 2007, p. 6), relaxed plain and kernel *k-means clustering* (Ding and He, 2004), and *squared-loss mutual information based clustering* (SMIC) (Sugiyama et al., 2011) as special limit cases. We demonstrate these claims in this section.

First of all, consider USC. The relaxed *RatioCut* problem can formulate USC from a graph cut point of view as

$$\min_{f \in \mathbb{R}^n} f^\top L_{\text{un}} f \qquad \text{s.t. } f \perp \mathbf{1}_n, \|f\|_2 = \sqrt{n} \tag{14}$$

when the number of clusters is two, where L_{un} is the *unnormalized graph Laplacian* (von Luxburg, 2007, pp. 10–11). Note that we can rewrite the primal problem of MVC-SL defined in (4) as

$$\min_{\boldsymbol{h} \in \mathbb{R}^n} -2\|\boldsymbol{h}\|_1/\gamma + \boldsymbol{h}^\top Q \boldsymbol{h} \qquad \text{s.t. } \|\boldsymbol{h}\|_2 = 1. \tag{15}$$

Optimizations (15) and (4) share exactly the same optimal solution with/without the class balance constraint $-b \leq \boldsymbol{h}^\top \mathbf{1}_n \leq b$, though (15) has an optimal objective value γ times smaller than (4)'s. Now, let $Q = L_{\text{un}} + \varepsilon I_n$ with arbitrarily chosen $\varepsilon > 0$ to make sure the positive definiteness of Q. Assume that f^* is the solution to (14), and $\boldsymbol{h}_m^*$ is the solution to (15) with Q specified as above, a class balance parameter $b = 0$, and a regularization parameter $\gamma_m = m$ given a natural number m. Subsequently, it is obvious that

$$\lim_{m \to \infty} \boldsymbol{h}_m^* = f^*/\sqrt{n},$$

and

$$\lim_{m\to\infty} -2\|\boldsymbol{h}_m^*\|_1/\gamma_m + \boldsymbol{h}_m^{*\top} Q \boldsymbol{h}_m^* = f^{*\top} L_{\text{un}} f^*/n + \varepsilon,$$

since $\|\boldsymbol{h}_m^*\|_1 \le \sqrt{n}\|\boldsymbol{h}_m^*\|_2 = \sqrt{n}$ and then $\lim_{m\to\infty} \|\boldsymbol{h}_m^*\|_1/\gamma_m = 0$. Therefore, USC may be viewed as a special limit case of MVC-SL, that is, a special case with the specification $Q = L_{\text{un}} + \varepsilon I_n$ of a limit case as $\gamma \to \infty$.

Remark 1 The motivation of $f \perp \mathbf{1}_n$ in USC is very different from $\boldsymbol{h}^\top \mathbf{1}_n = 0$ in MVC-SL for class balancing. When L_{un} is constructed from a fully connected similarity graph, the constraint $f \perp \mathbf{1}_n$ means that the feasible region of optimization (14) is in a space spanned by all eigenvectors of L_{un} except the trivial eigenvector $\mathbf{1}_n$. Note that $\boldsymbol{h}^\top \mathbf{1}_n = 0$ just asks for strictly balanced soft responses and is not equivalent to $\text{sign}(\boldsymbol{h})^\top \mathbf{1}_n = 0$ that demands strictly balanced cluster assignments.

On the other hand, continuous solutions to the relaxations of k-means clustering (MacQueen, 1967; Hartigan and Wong, 1979) and kernel k-means clustering (Girolami, 2002) can be obtained by principle component analysis (PCA) and kernel PCA, respectively (Zha et al., 2002; Ding and He, 2004). Now, let $Q = \varepsilon I_n - C_n K C_n$ with arbitrarily chosen $\varepsilon > \|C_n K C_n\|_2$, where K is the *kernel matrix*, $C_n = I_n - \mathbf{1}_n \mathbf{1}_n^\top/n$ is the centering matrix, and $\|\cdot\|_2$ here means the spectral norm (which is also known as the operator norm induced by the ℓ_2-norm) of a matrix. As a result,

$$\lim_{m\to\infty} \boldsymbol{h}_m^* = \boldsymbol{v}^*,$$

and

$$\lim_{m\to\infty} -2\|\boldsymbol{h}_m^*\|_1/\gamma_m + \boldsymbol{h}_m^{*\top} Q \boldsymbol{h}_m^* = \varepsilon - \boldsymbol{v}^* C_n K C_n \boldsymbol{v}^*,$$

where $\boldsymbol{h}_m^*$ is the solution to (15) with Q specified as above and $\gamma_m = m$, and $\boldsymbol{v}^*$ is the solution to the relaxed kernel k-means clustering (Ding and He, 2004, Theorem 3.5). In addition, if $\mathcal{X} \subset \mathbb{R}^d$ and $X \in \mathbb{R}^{n\times d}$ is the design matrix, we will have

$$\lim_{m\to\infty} \boldsymbol{h}_m'^* = \boldsymbol{v}'^*,$$

and

$$\lim_{m\to\infty} -2\|\boldsymbol{h}_m'^*\|_1/\gamma_m + \boldsymbol{h}_m'^{*\top} Q \boldsymbol{h}_m'^* = \varepsilon - \boldsymbol{v}'^* C_n X X^\top C_n \boldsymbol{v}'^*,$$

where $\boldsymbol{h}_m'^*$ is the solution to (15) with $Q = \varepsilon I_n - C_n X X^\top C_n$, $\varepsilon > \|C_n X X^\top C_n\|_2$ and $\gamma_m = m$, and $\boldsymbol{v}'^*$ is the solution to the relaxed plain k-means clustering (Ding and He, 2004, Theorem 2.2). In other words, plain k-means clustering and kernel k-means clustering after certain relaxations are special limit cases of MVC-SL.[5]

Similarly to USC and two k-means clustering, the optimization problem of the binary SMIC is another special limit case of MVC-SL. It involves the maximization of the following *squared-loss mutual information* approximator

$$\max_{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2\in\mathbb{R}^n} \frac{1}{n}\sum_{y=1}^{2} \boldsymbol{\alpha}_y^\top K^2 \boldsymbol{\alpha}_y - \frac{1}{2} \tag{16}$$

5. When considering k-means algorithms that are referred to as certain iterative clustering algorithms rather than clustering models, by no means they can be special limit cases of MVC-SL.

under an orthonormal constraint of $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$, where $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ are model parameters of posterior probabilities and K is the kernel matrix. The optimal solutions to (16) can be obtained through

$$\boldsymbol{\alpha}_1^* = \mathop{\arg\max}_{\boldsymbol{\alpha}\in\mathbb{R}^n} \boldsymbol{\alpha}^\top K^2 \boldsymbol{\alpha} \qquad \text{s.t. } \|\boldsymbol{\alpha}\|_2 = 1, \tag{17}$$

$$\boldsymbol{\alpha}_2^* = \mathop{\arg\max}_{\boldsymbol{\alpha}\in\mathbb{R}^n} \boldsymbol{\alpha}^\top K^2 \boldsymbol{\alpha} \qquad \text{s.t. } \boldsymbol{\alpha}\perp\boldsymbol{\alpha}_1^*, \|\boldsymbol{\alpha}\|_2 = 1. \tag{18}$$

Now, let $Q = \varepsilon I_n - K^2$ with arbitrarily chosen $\varepsilon > \|K\|_2^2$. We could then know

$$\lim_{m\to\infty} \boldsymbol{h}_{1,m}^* = \boldsymbol{\alpha}_1^*,$$

and

$$\lim_{m\to\infty} -2\|\boldsymbol{h}_{1,m}^*\|_1/\gamma_m + \boldsymbol{h}_{1,m}^{*\top} Q \boldsymbol{h}_{1,m}^* = \varepsilon - \boldsymbol{\alpha}_1^{*} K^2 \boldsymbol{\alpha}_1^*,$$

where $\boldsymbol{h}_{1,m}^*$ is the solution to (15) with Q specified as above and $\gamma_m = m$. Likewise,

$$\lim_{m\to\infty} \boldsymbol{h}_{2,m}^* = \boldsymbol{\alpha}_2^*,$$

and

$$\lim_{m\to\infty} -2\|\boldsymbol{h}_{2,m}^*\|_1/\gamma_m + \boldsymbol{h}_{2,m}^{*\top} Q \boldsymbol{h}_{2,m}^* = \varepsilon - \boldsymbol{\alpha}_2^{*} K^2 \boldsymbol{\alpha}_2^*,$$

where $\boldsymbol{h}_{2,m}^*$ is the solution to (15) with Q specified as above, $\gamma_m = m$, and a constraint $\boldsymbol{h}^\top \boldsymbol{h}_{1,m}^* = 0$.

Remark 2 After optimizing (17) and (18), SMIC adopts the post-processing that encloses $\boldsymbol{\alpha}_1^*$ and $\boldsymbol{\alpha}_2^*$ into posterior probabilities and enables the out-of-sample clustering ability for any $x \in \mathcal{X}$ even if $x \notin X_n$ (Sugiyama et al., 2011), while MVC-SL can use

$$\boldsymbol{h}^* = \boldsymbol{\alpha}_1^* \operatorname{sign}(\mathbf{1}_n^\top \boldsymbol{\alpha}_1^*) - \boldsymbol{\alpha}_2^* \operatorname{sign}(\mathbf{1}_n^\top \boldsymbol{\alpha}_2^*)$$

as the optimal soft response vector since there are just two clusters.

5. Finite Sample Stability

The stability of the resulting clusters is important for clustering models whose non-convex primal problems are solved by randomized algorithms (e.g., MVC-SL and k-means clustering) rather than relaxed to convex dual problems (e.g., MVC-HL and MMC). To this end, we investigate the finite sample stability of the primal problem of MVC-SL in this section.

In the following, we presume that we are always able to find a locally optimal solution to optimization (4) accurately. Under this presumption, we prove that the instability is resulted from the symmetry of data samples: As long as the input matrix Q satisfies some asymmetry condition, we could obtain the same data partition based on different locally optimal solutions, and consequently the non-convex optimization of MVC-SL seems convex.

5.1 Definitions

Definition 3 *The Hamming clustering distance for two n-dimensional soft response vectors $\boldsymbol{h}$ and $\boldsymbol{h}'$ is defined as*

$$d_{\mathcal{H}}(\boldsymbol{h}, \boldsymbol{h}') := \frac{1}{2} \min(\|\operatorname{sign}(\boldsymbol{h}) + \operatorname{sign}(\boldsymbol{h}')\|_1, \|\operatorname{sign}(\boldsymbol{h}) - \operatorname{sign}(\boldsymbol{h}')\|_1).$$

When measuring the difference of two binary clusterings, $d_{\mathcal{H}}(\boldsymbol{h},\boldsymbol{h}')$ is always a natural number smaller than $n/2$, since $\|\operatorname{sign}(\boldsymbol{h})+\operatorname{sign}(\boldsymbol{h}')\|_1+\|\operatorname{sign}(\boldsymbol{h})-\operatorname{sign}(\boldsymbol{h}')\|_1=2n$.

Definition 4 (Irreducibility) *A sample x_i is isolated in X_n, if $Q_{i,i}>0$ and $\forall j\neq i, Q_{i,j}=0$. A set of samples X_n is irreducible, if no sample is isolated in X_n; otherwise X_n is reducible.*

The idea behind the irreducibility of X_n is simple: If x_i is isolated, we cannot decide its cluster based on the information contained in Q no matter what binary clustering algorithm is used, unless we assign x_i to one cluster and $X_n\setminus x_i$ to the other cluster. We would like to remove such isolated samples and reduce the clustering of X_n to a better-defined problem.

Next we define two symmetry concepts, the submatrix-information- (SI- for short) symmetry in Definition 5 and the axisymmetry in Definition 7. SI-asymmetry is a part of the sufficient condition for finite sample stability, and axisymmetry is a part of the sufficient condition for instability. The relationship of irreducibility, axisymmetry and SI-symmetry will be proved in Theorem 10.

Definition 5 (Submatrix-Information-Symmetry) *A set of samples X_n is submatrix-information-symmetric, if there exist $\{\delta_1,\ldots,\delta_n\}\in\{-1,+1\}^n$ and nonempty $\mathcal{K}\subsetneq\{1,\ldots,n\}$ such that*

$$\sum_{i\in\mathcal{K},j\notin\mathcal{K},\delta_i=\delta_j}Q_{i,j}=\sum_{i\in\mathcal{K},j\notin\mathcal{K},\delta_i\neq\delta_j}Q_{i,j}. \tag{19}$$

Otherwise, X_n is submatrix-information-asymmetric.[6]

Remark 6 It is clear that

$$\sum_{i\in\mathcal{K},j\notin\mathcal{K},\delta_i=\delta_j}Q_{i,j}=\sum_{i\in\mathcal{K},j\notin\mathcal{K}}\delta_i\delta_jQ_{i,j},$$
$$\sum_{i\in\mathcal{K},j\notin\mathcal{K},\delta_i\neq\delta_j}Q_{i,j}=-\sum_{i\in\mathcal{K},j\notin\mathcal{K}}\delta_i\delta_jQ_{i,j},$$

and thus Equation (19) is equivalent to

$$\left(\sum_{k\in\mathcal{K}}\delta_k\boldsymbol{e}_k\right)^{\top}Q\left(\sum_{k\notin\mathcal{K}}\delta_k\boldsymbol{e}_k\right)=0, \tag{20}$$

where $\{\boldsymbol{e}_1,\ldots,\boldsymbol{e}_n\}$ is a standard basis of $\mathbb{R}^n$. From now on, we may use Equation (20) as the condition to check SI-symmetry or SI-asymmetry for convenience.

Intuitively, the SI-symmetry of X_n says that Q has a submatrix containing the same amount of similarity and dissimilarity information. More specifically, both $\{\delta_1,\ldots,\delta_n\}$ and $\mathcal{K}$ are valid partitions of X_n, though they have different representations and functions. The partition $\{\delta_1,\ldots,\delta_n\}$ is a reference for similarity and dissimilarity, and based on this partition, we categorize the information $Q_{i,j}$ between x_i and x_j into similarity information if $\delta_i=\delta_j$ or dissimilarity information if $\delta_i\neq\delta_j$. On the other hand, we divide Q into four parts $Q[i\in\mathcal{K};j\in\mathcal{K}]$, $Q[i\in\mathcal{K};j\notin\mathcal{K}]$, $Q[i\notin\mathcal{K};j\in\mathcal{K}]$ and $Q[i\notin\mathcal{K};j\notin\mathcal{K}]$ according to the partition $\mathcal{K}$. The SI-symmetry of X_n shown in Equation (19)

6. Strictly speaking, saying that X_n is SI-symmetric is a bit abuse of terminology. In formal mathematical terminology, an object is symmetric with respect to some operation, if this operation, when applied to the object, preserves certain property. For example, in the axisymmetry, the object is X_n, the operation is ϕ and the property is Q. However, in the SI-symmetry, the object is a set of two vectors $\{\sum_{k\in\mathcal{K}}\delta_k\boldsymbol{e}_k,\sum_{k\notin\mathcal{K}}\delta_k\boldsymbol{e}_k\}$, the operation is replacing I_n with Q, and the property is the orthogonality (preserved from the standard orthogonality to the Q-orthogonality).

indicates that the submatrix $Q[i \in \mathcal{K}; j \notin \mathcal{K}]$ (and likewise $Q[i \notin \mathcal{K}; j \in \mathcal{K}]$) contains the same amount of similarity information (i.e., the left-hand side) and dissimilarity information (i.e., the right-hand side).

When X_n is SI-symmetric, we could easily find two feasible solutions to optimization (4), such that they would induce different partitions of X_n but share the same value of the objective function. To see this, let

$$\boldsymbol{h}_+ = \frac{\sum_{k\in\mathcal{K}} \delta_k \boldsymbol{e}_k + \sum_{k\notin\mathcal{K}} \delta_k \boldsymbol{e}_k}{\sqrt{n}},$$
$$\boldsymbol{h}_- = \frac{\sum_{k\in\mathcal{K}} \delta_k \boldsymbol{e}_k - \sum_{k\notin\mathcal{K}} \delta_k \boldsymbol{e}_k}{\sqrt{n}}.$$

It is easy to verify that $\|\boldsymbol{h}_\pm\|_2 = 1$, $\|\boldsymbol{h}_\pm\|_1 = \sqrt{n}$ and $d_{\mathcal{H}}(\boldsymbol{h}_+, \boldsymbol{h}_-) \geq 1$. Moreover,

$$\boldsymbol{h}_+^\top Q \boldsymbol{h}_+ = \boldsymbol{h}_+^\top Q \boldsymbol{h}_+ - (\boldsymbol{h}_+ + \boldsymbol{h}_-)^\top Q (\boldsymbol{h}_+ - \boldsymbol{h}_-) = \boldsymbol{h}_-^\top Q \boldsymbol{h}_-,$$

where we used $(\boldsymbol{h}_+ + \boldsymbol{h}_-)^\top Q (\boldsymbol{h}_+ - \boldsymbol{h}_-) = 0$ by the condition Equation (20) of SI-symmetry. However, $\boldsymbol{h}_+$ and $\boldsymbol{h}_-$ are not necessarily locally optimal solutions to optimization (4), and maybe no locally optimal solution could induce the same partition with $\boldsymbol{h}_+$ or $\boldsymbol{h}_-$. The real reason for finite sample instability is the axisymmetry of data samples defined below.

Definition 7 (Axisymmetry) *A set of samples X_n is axisymmetric, if there exists a permutation ϕ of $\{1,\ldots,n\}$, such that*

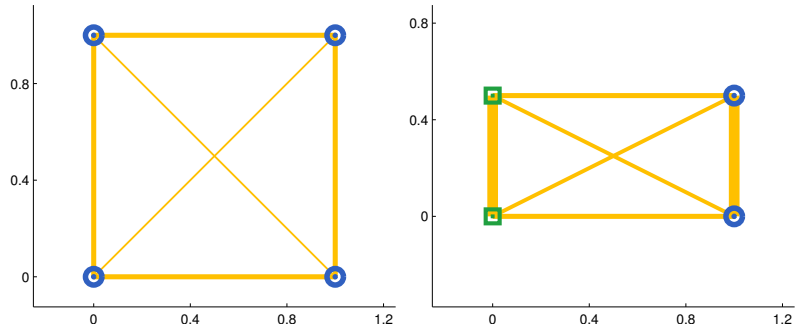
1. $\exists i \in \{1,\ldots,n\}, \phi(i) \neq i$;
2. $\forall i \in \{1,\ldots,n\}, \phi^{-1}(i) = \phi(i)$;
3. $\forall 1 \leq i,j \leq n, Q_{i,j} = Q_{\phi(i),\phi(j)}$.

The first property says that the permutation ϕ cannot be the identical mapping: It allows some, but not all, sample x_i mapped to itself. The second property requires that those samples mapped to others are all paired. In other words, X_n is separated into two types of disjoint subsets according to ϕ, and they are either cardinality one (i.e., $\{x_i \mid \phi(i) = i\}$) or two (i.e., $\{x_i, x_{\phi(i)} \mid \phi(i) \neq i\}$), but no greater cardinality. The third property guarantees that Q is ϕ-invariant, or equivalently the samples in the subset $\{x_i, x_{\phi(i)} \mid \phi(i) \neq i\}$ cannot be distinguished by all other subsets $\{x_j \mid \phi(j) = j, j \neq i\}$ or $\{x_j, x_{\phi(j)} \mid \phi(j) \neq j, j \neq i, j \neq \phi(i)\}$ based on the information contained in Q, so we can exchange x_i and $x_{\phi(i)}$ freely without modifying Q.

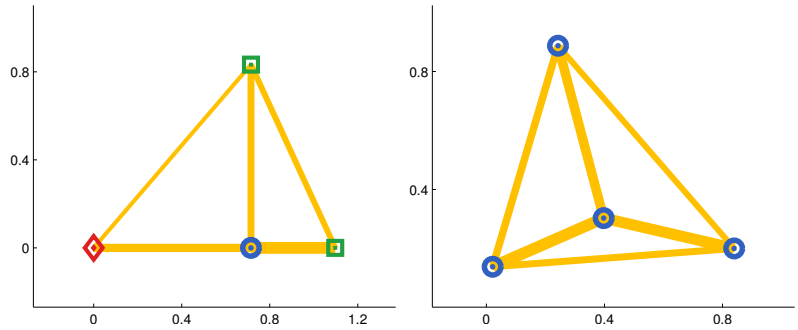
The axisymmetry of X_n in terms of Q is equivalent to the geometric axisymmetry of X_n in $\mathcal{X}$, if $\mathcal{X} \subset \mathbb{R}^d$ and Q is a matrix induced from the Euclidean distance such as a Gaussian kernel matrix. For example, as shown in Figure 2,

$$X_4 = \{(0,0),(1,0),(1,1),(0,1)\},$$
$$X_4' = \{(0,0),(1,0),(1,0.5),(0,0.5)\}$$

are axisymmetric both in $\mathbb{R}^2$ and in terms of Q if Q is a Gaussian kernel matrix, regardless of the kernel width. The permutation ϕ for X_4' could be $\{(1,2),(3,4)\}$, $\{(1,3),(2,4)\}$ or $\{(1,4),(2,3)\}$, and besides them, ϕ for X_4 could also be $\{(1),(3),(2,4)\}$ or $\{(1,3),(2),(4)\}$. We can identify an



(a) X_4 (b) X_4'



(c) X_4'' (d) Random

Figure 2: Four-point sets that are typical in the theory of finite sample stability. Gaussian similarities ($\sigma = 1/\sqrt{2}$) between nodes are visualized by the line thickness of edges. All sets in this figure are irreducible. X_4 in panel (a) is axisymmetric and SI-symmetric. X_4' in (b) is axisymmetric, SI-symmetric, anisotropic, and has a unique best partition. X_4'' in (c) is very special: It is anisotropic, *SI-symmetric but not axisymmetric*, since the similarity of the diamond and circle equals the sum of the similarities of the diamond and squares. A random set would be anisotropic and SI-asymmetric with high probability.

axis of symmetry geometrically in $\mathbb{R}^d$: It must pass through either x_i if $\phi(i) = i$ or $(x_i + x_{\phi(i)})/2$ if $\phi(i) \neq i$ for $i = 1, \ldots, n$. This is why we call such a property axisymmetry.

Generally speaking, the concepts of axisymmetry and SI-symmetry almost coincide, if Q is a Gaussian kernel matrix or the corresponding graph Laplacian matrix. While it is possible to deliberately construct counter-examples that are SI-symmetric but not axisymmetric, it is improbable to meet a counter-example in practice. For instance, as illustrated in panel (c) of Figure 2,

$$\begin{aligned} X_4'' &= \{(0,0), (\sqrt{\ln(5/3)}, 0), (\sqrt{\ln(10/3)}, 0), (\sqrt{\ln(5/3)}, \sqrt{\ln 2})\} \\ &\approx \{(0,0), (0.7147, 0), (1.0973, 0), (0.7147, 0.8326)\} \end{aligned}$$

is SI-symmetric but not axisymmetric in terms of Gaussian kernel matrix Q when $\sigma = 1/\sqrt{2}$, yet X_4'' is SI-asymmetric whenever $\sigma \neq 1/\sqrt{2}$.

Definition 8 (Anisotropy) *A set of samples X_n is anisotropic, if Q has n distinct eigenvalues.*

The anisotropy of X_n is the other part of the sufficient condition for finite sample stability. The name comes from a geometric interpretation of the ellipsoid $\mathcal{E}(\mathcal{H}_Q)$: All its principal axes achieve distinct lengths when Q has distinct eigenvalues, and thus $\mathcal{E}(\mathcal{H}_Q)$ is anisotropic and not rotatable. The concepts of anisotropy and axisymmetry are not complementary, since they concern different aspects of different objects, that is, the rotation of $\mathcal{E}(\mathcal{H}_Q)$ vs. the reflection of X_n. In Figure 2, X_4 is axisymmetric, X_4' is axisymmetric and anisotropic, and most random sets are just anisotropic. There might be X_n neither axisymmetric nor anisotropic. Nonetheless, when considering the more general SI-symmetry and certain families of Q such as Gaussian kernel matrices, X_n is anisotropic as long as it is SI-asymmetric.

All definitions have been discussed. The theoretical results will be presented next.

5.2 Theoretical Results

The following lemma will be used in Theorems 11 and 13. All proofs are provided in Appendix A.

Lemma 9 *Let X_n be an irreducible set of samples, $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the normalized eigenvectors of Q, and $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$ be a standard basis of $\mathbb{R}^n$. Then, $\forall i, j \in \{1, \ldots, n\}$, $\boldsymbol{v}_i \neq \pm \boldsymbol{e}_j$.*

The following two theorems describe the relationship between the properties defined above.

Theorem 10 *A set of samples X_n is SI-symmetric, if it is reducible or axisymmetric.*

Theorem 11 *If X_n is an SI-asymmetric set of samples, and there exists $\kappa > 0$ such that $Q_{1,1} = Q_{2,2} = \cdots = Q_{n,n} = \kappa$, then X_n is anisotropic.*

We are ready to deliver our main theorems. To begin with, given a constant η, we define (recall the assumption that $\|\boldsymbol{h}\|_1$ is differentiable thanks to the non-sparsity of $\boldsymbol{h}$)

$$G(\boldsymbol{h}) := \gamma \boldsymbol{h}^\top Q \boldsymbol{h} - \eta \|\boldsymbol{h}\|_2^2 - 2\|\boldsymbol{h}\|_1,$$
$$g(\boldsymbol{h}) := \frac{1}{2} \nabla G(\boldsymbol{h}) = \gamma Q \boldsymbol{h} - \eta \boldsymbol{h} - \operatorname{sign}(\boldsymbol{h}).$$

Theorem 12 (Twin Minimum Theorem) *Assume that $n > 2$, X_n is an axisymmetric set of samples, ϕ is the corresponding permutation, and $I = \{\{i, \phi(i)\} \mid \phi(i) \neq i\}$ is the index set of those paired samples given ϕ. For every minimum $\boldsymbol{h}^*$ of optimization* (4)*, if*

1. *$\forall i, [\boldsymbol{h}^*]_i \neq 0$, and*
2. *$\exists \{i, \phi(i)\} \in I, [\boldsymbol{h}^*]_{\phi(i)} [\boldsymbol{h}^*]_i < 0$,*

then $\boldsymbol{h}^*$ has a twin minimum $\boldsymbol{h}^\star$ satisfying $G(\boldsymbol{h}^\star) = G(\boldsymbol{h}^*)$ and $d_{\mathcal{H}}(\boldsymbol{h}^*, \boldsymbol{h}^\star) \geq 1$. The only exception is

$$\forall i \in \{1, \ldots, n\}, [\boldsymbol{h}^*]_{\phi(i)} [\boldsymbol{h}^*]_i < 0.$$

In order to explain the implication of Theorem 12, let us recall X_4 and X_4' shown in Figure 2. There are many twin minima when considering the perfectly symmetric X_4, but it is same even for those convex relaxations of MMC due to the post-processing. On the other hand, X_4' illustrates an exception: While X_4 allows $\phi(i) = i$, this is impossible for X_4'. More specifically, any minimum $\boldsymbol{h}^*$

corresponding to partition $(+1,-1,-1,+1)$ has no twin minimum, since ϕ can be $\{(1,2),(3,4)\}$, $\{(1,3),(2,4)\}$ or $\{(1,4),(2,3)\}$ for X_4', and

$$\forall \phi, (\exists i, [\boldsymbol{h}^*]_{\phi(i)}[\boldsymbol{h}^*]_i < 0) \rightarrow (\forall i, [\boldsymbol{h}^*]_{\phi(i)}[\boldsymbol{h}^*]_i < 0).$$

It suggests that if we permute $\boldsymbol{h}^*$ according to ϕ, then $\text{sign}(\boldsymbol{h}^\star) = \pm\text{sign}(\boldsymbol{h}^*)$ is the same partition and thus $d_{\mathcal{H}}(\boldsymbol{h}^*, \boldsymbol{h}^\star) = 0$. Another minimum $\boldsymbol{h}$ that corresponds to $(+1,+1,-1,-1)$ and satisfies $d_{\mathcal{H}}(\boldsymbol{h}^*, \boldsymbol{h}) \geq 1$ should have $G(\boldsymbol{h}) > G(\boldsymbol{h}^*)$. In a word, local minima that correspond to different partitions for X_4' are not equally good and the best partition is still unique, as illustrated in panel (b) of Figure 2. The genuine instability emerges only when the best partition is not unique, like X_4 in panel (a) of Figure 2.

Theorem 13 (Equivalent Minima Theorem) *All minima of optimization* (4) *are equivalent with respect to $d_{\mathcal{H}}$, provided that*

1. *X_n is SI-asymmetric;*
2. *X_n is anisotropic.*

By combining Theorem 11 and Theorem 13, we have a corollary immediately.

Corollary 14 *All minima of optimization* (4) *are equivalent with respect to $d_{\mathcal{H}}$, provided that*

1. *X_n is SI-asymmetric;*
2. *There exists $\kappa > 0$ such that $Q_{1,1} = Q_{2,2} = \cdots = Q_{n,n} = \kappa$.*

To sum up, if Q has the two properties listed above, different locally optimal solutions to optimization (4) would ideally induce the same data partition. Nevertheless, the output of the algorithm is not in the same form as the solution to optimization (4), since the variable η has been introduced, and we cannot foresee its optimal value when we analyze the original model. Spectral clustering is consistent (von Luxburg et al., 2008), but it has a similar problem in finite sample stability, that is, when the graph Laplacian has distinct eigenvalues and the unique spectral decomposition leads to a stable spectral embedding, the following k-means clustering can still introduce high instability due to the non-convex nature of the distortion function.

Remark 15 We rely on a Karush-Kuhn-Tucker stationarity condition $g(\boldsymbol{h}^*) = \mathbf{0}_n$ in the proofs of Theorems 12 and 13, where $\boldsymbol{h}^*$ is the optimal solution to (4). Actually, the objective function of (4) usually has a non-zero derivative and the objective function of (3) always has a non-zero derivative in their feasible regions. Therefore, we introduce the functions $G(\boldsymbol{h})$ and $g(\boldsymbol{h})$ to analyze MVC-SL from a theoretical point of view. In MVC-SL, Equation (7) comes from the least-square fitting of Equation (8), and if $t \rightarrow \infty$, we will have $\boldsymbol{p}_t^* \rightarrow \mathbf{0}_n$ and then Equation (8) will turn into $g(\boldsymbol{h}^*) = \lim_{t\rightarrow\infty} g(\boldsymbol{h}_t) = \mathbf{0}_n$.

6. Clustering Error Bound

In this section, we derive a clustering error bound for MVC-SL based on *transductive Rademacher complexity* (El-Yaniv and Pechyony, 2009).

It is extremely difficult, if possible, to evaluate clustering methods in an objective and domain-independent manner (von Luxburg et al., 2012). However, when the goals and interests are clear, it

makes sense to evaluate clustering results using classification benchmark data sets, where the class structure coincides with the desired cluster structure according to the goals and interests.

In real-world applications, we often find some experts to cluster a small portion $X_{n'}$ of X_n where $n' < n$ according to their professional knowledge, test a lot of clustering methods with a lot of similarity measures, see their agreement with given clustering of $X_{n'}$, and eliminate those low agreement methods. This procedure may be viewed as propagating the knowledge of experts from $X_{n'}$ to X_n.

Here, we derive a data-dependent clustering error bound to guarantee the quality of this propagation of knowledge. The key technique is transductive Rademacher complexity for deriving data-dependent transductive error bounds. To begin with, we follow El-Yaniv and Pechyony (2009) for the definition of transductive Rademacher complexity:

Definition 16 *Fix positive integers m and u. Let $\mathcal{H} \subseteq \mathbb{R}^{m+u}$ be a hypothesis space, $p \in [0, 1/2]$ be a parameter, and $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_{m+u})^\top$ be a vector of independent and identically distributed random variables, such that*

$$\sigma_i := \begin{cases} +1 & \text{with probability} \quad p, \\ -1 & \text{with probability} \quad p, \\ 0 & \text{with probability} \quad 1-2p. \end{cases}$$

Then, the transductive Rademacher complexity of $\mathcal{H}$ with parameter p is defined as

$$\mathcal{R}_{m+u}(\mathcal{H}, p) := \left(\frac{1}{m} + \frac{1}{u}\right) \mathbb{E}_{\boldsymbol{\sigma}} \left\{ \sup_{\boldsymbol{h} \in \mathcal{H}} \boldsymbol{\sigma}^\top \boldsymbol{h} \right\}.$$

For the sake of comparison, we give a definition of inductive Rademacher complexity following El-Yaniv and Pechyony (2009).[7]

Definition 17 *Let $p(x)$ be a probability density on $\mathcal{X}$, and $X_n = \{x_1, \ldots, x_n\}$ be a set of independent observations drawn from $p(x)$. Let $\mathcal{H}$ be a class of functions from $\mathcal{X}$ to $\mathbb{R}$, and $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)^\top$ be a vector of independent and identically distributed random variables, such that*

$$\sigma_i := \begin{cases} +1 & \text{with probability} \quad 1/2, \\ -1 & \text{with probability} \quad 1/2. \end{cases}$$

The empirical Rademacher complexity of $\mathcal{H}$ conditioned on X_n is

$$\widehat{\mathcal{R}}_u^{(ind)}(\mathcal{H}) := \frac{2}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left\{ \sup_{h \in \mathcal{H}} \boldsymbol{\sigma}^\top \boldsymbol{h} \mid X_n \right\},$$

where $\boldsymbol{h} = (h(x_1), \ldots, h(x_n))^\top$, and the inductive Rademacher complexity of $\mathcal{H}$ is

$$\mathcal{R}_u^{(ind)}(\mathcal{H}) := \mathbb{E}_{X_n} \left\{ \widehat{\mathcal{R}}_u^{(ind)}(\mathcal{H}) \right\}.$$

The transductive Rademacher complexity of $\mathcal{H}$ is an empirical quantity that depends only on p. Given fixed X_n, we have $\mathcal{R}_{m+u}(\mathcal{H}) = 2\widehat{\mathcal{R}}_{m+u}^{(ind)}(\mathcal{H})$ when $p = 1/2$ and $m = u$.[8] Whenever $p < 1/2$,

7. Albeit there are many definitions of Rademacher complexity, for example, Koltchinskii (2001), Bartlett and Mendelson (2002), Meir and Zhang (2003) and Bousquet et al. (2004), they are similar and conceptually equivalent.
8. A class of functions conditioned on fixed data is equivalent to a hypothesis space of soft response vectors.

some Rademacher variables will attain zero values and reduce the complexity. We simply consider $p_0 = mu/(m+u)^2$ and abbreviate $\mathcal{R}_{m+u}(\mathcal{H}, p_0)$ to $\mathcal{R}_{m+u}(\mathcal{H})$ as El-Yaniv and Pechyony (2009) in Lemma 18 and Theorem 19, though these theoretical results hold for all $p > p_0$ since $\mathcal{R}_{m+u}(\mathcal{H}, p)$ is monotonically increasing with p. Please refer to El-Yaniv and Pechyony (2009) for the detailed discussions about transductive Rademacher complexity.

Lemma 18 *Let $\mathcal{H}'_Q$ be the set of all possible $\boldsymbol{h}$ returned by Algorithm 1 for the given Q, η^* be the optimal η when Algorithm 1 stops,*

$$\mu = \sup_{\boldsymbol{h} \in \mathcal{H}'_Q} \operatorname{sign}(\boldsymbol{h})^\top (\gamma Q - \eta^* I_n)^{-1} \operatorname{sign}(\boldsymbol{h}),$$

and $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of Q. Then, for the transductive Rademacher complexity of $\mathcal{H}'_Q$, the following upper bound holds for any integer $n' = 1, 2, \ldots, n-1$,

$$\mathcal{R}_u(\mathcal{H}'_Q) \leq \sqrt{\frac{2}{n'(n-n')}} \min \left\{ \sqrt{n}, \left(\sum_{i=1}^{n} \frac{n}{(\gamma\lambda_i - \eta^*)^2} \right)^{1/2}, \left(\sum_{i=1}^{n} \frac{\mu}{\gamma\lambda_i - \eta^*} \right)^{1/2} \right\}.$$

The proof of Lemma 18 can be found in Appendix B. By Lemma 18 together with Theorem 2 of El-Yaniv and Pechyony (2009), we can immediately obtain the clustering error bound:

Theorem 19 *Assume that $\boldsymbol{y}^*$ is the ground truth partition of X_n, and $\mathcal{L}$ is a random set of size n' chosen uniformly from the set $\{\mathcal{L} \mid \mathcal{L} \subset \{1, \ldots, n\}, \#\mathcal{L} = n'\}$. Let $\ell(z) = \min(1, \max(0, 1-z))$ for $z \in \mathbb{R}$ be the surrogate loss, $\mathcal{H}'_Q$ be the set of all possible $\boldsymbol{h}$ returned by Algorithm 1 for the given Q, η^* be the optimal η when Algorithm 1 stops,*

$$\mu = \sup_{\boldsymbol{h} \in \mathcal{H}'_Q} \operatorname{sign}(\boldsymbol{h})^\top (\gamma Q - \eta^* I_n)^{-1} \operatorname{sign}(\boldsymbol{h}),$$

$\lambda_1, \ldots, \lambda_n$ be the eigenvalues of Q, and $c_0 = \sqrt{32(1+\ln 4)/3}$. For any $\boldsymbol{h} \in \mathcal{H}'_Q$, with probability at least $1-\delta$ over the choice of $\mathcal{L}$, we have

$$\begin{aligned} d_{\mathcal{H}}(\boldsymbol{h}, \boldsymbol{y}^*) \leq{}& \frac{n}{n'} \min \left\{ \sum_{i \in \mathcal{L}} \ell([\boldsymbol{h}]_i [\boldsymbol{y}^*]_i), \sum_{i \in \mathcal{L}} \ell(-[\boldsymbol{h}]_i [\boldsymbol{y}^*]_i) \right\} \\ &+ \frac{c_0 n}{\sqrt{n'}} + \sqrt{\frac{2n^2(n-n')^2}{n'(2n-1)(2n-2n'-1)} \ln(1/\delta)} \\ &+ \sqrt{\frac{2(n-n')}{n'}} \min \left\{ \sqrt{n}, \left(\sum_{i=1}^{n} \frac{n}{(\gamma\lambda_i - \eta^*)^2} \right)^{1/2}, \left(\sum_{i=1}^{n} \frac{\mu}{\gamma\lambda_i - \eta^*} \right)^{1/2} \right\}. \end{aligned} \tag{21}$$

There are four terms in the right-hand side of inequality (21). The first term is a measure of the clustering error on $X_{n'} = \{x_i \mid i \in \mathcal{L}\}$ by the surrogate loss times the ratio n/n'. More specifically, we would like to select a proper similarity measure via given clustering $\{[\boldsymbol{y}^*]_i \mid i \in \mathcal{L}\}$ to make the error on $X_{n'}$ as small as possible, under the assumption that the error rates on $X_{n'}$ and X_n should be close for a fixed similarity measure (the given $\{[\boldsymbol{y}^*]_i \mid i \in \mathcal{L}\}$ are not used for training). The second term depends only on n and n', i.e., the sizes of the whole set and the clustered subset. Besides n and n', the third term further depends on the significance level δ, as in common error bounds. The last term

is the upper bound of $(n-n')\mathcal{R}_u(\mathcal{H}'_Q)$, which carries out the complexity control of $\mathcal{H}'_Q$ implicitly: The smaller the value of $\mathcal{R}_u(\mathcal{H}'_Q)$ is, the more confident we are that $d_{\mathcal{H}}(\boldsymbol{h},\boldsymbol{y}^*)$ would be small, if the error on $X_{n'}$ is small. When considering the average clustering error measured by $d_{\mathcal{H}}(\boldsymbol{h},\boldsymbol{y}^*)/n$, the order of the error bound is $O(1/\sqrt{n'})$ since the second term dominates the third and fourth terms.

Remark 20 Our problem setting is equivalent to neither semi-supervised clustering nor transductive classification: We do not reveal any labels to the clustering algorithm in Theorem 19; instead, a set of randomly chosen labels are revealed to an evaluator who then returns the evaluation of the quality of any possible partition generated by the algorithm. We can use the theory of transductive Rademacher complexity to derive a clustering error bound for Algorithm 1, since it can be viewed as a transductive algorithm that ignores all revealed labels.

7. Related Works

In this section, we review related works and qualitatively compare the proposed MVC with them.

7.1 Maximum Margin Clustering

Among existing clustering methods, *maximum margin clustering* (MMC) is closest to MVC. Both of them originate from statistical learning theory, but their geneses and underlying criteria are still different: The primal problems of all MMC adopt a regularizer $\|\boldsymbol{w}\|_2^2$ from the margin, while MVC relies on the regularizer $V(\boldsymbol{h})$ in Equation (1) from the volume. The hypothesis shared by all MMC is the hyperplane for induction, while the hypothesis in MVC is the soft response vector for transduction. The latter is more natural, since clustering is more transductive than inductive.

The family of MMC algorithms was initiated by Xu et al. (2005). It follows the support vector machine (SVM) (Boser et al., 1992; Cortes and Vapnik, 1995) and its hard-margin version can be formulated as

$$\min_{\boldsymbol{y}\in\{-1,+1\}^n} \min_{\boldsymbol{w},\boldsymbol{\xi}} \ \|\boldsymbol{w}\|_2^2 \\ \text{s.t. } y_i\boldsymbol{w}^\top x_i \geq 1, \quad i=1,\ldots,n.$$

The value of $y_i\boldsymbol{w}^\top x_i$ is called the functional margin of (x_i,y_i), whereas the value of $y_i\boldsymbol{w}^\top x_i/\|\boldsymbol{w}\|_2$ is called the geometric margin of (x_i,y_i). MMC can maximize the geometric margin of all $x_i \in X_n$ over $\boldsymbol{y}\in\{-1,+1\}^n$ by minimizing $\|\boldsymbol{w}\|_2$ and requiring the minimal functional margin to be one simultaneously. Likewise, the primal problem of the soft-margin MMC is

$$\min_{\boldsymbol{y}\in\{-1,+1\}^n} \min_{\boldsymbol{w},\boldsymbol{\xi}} \ \|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^n \xi_i \\ \text{s.t. } y_i\boldsymbol{w}^\top x_i \geq 1-\xi_i, \xi_i \geq 0, \quad i=1,\ldots,n,$$

where $C > 0$ is a regularization parameter, and $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)^\top$ is a vector of slack variables. Then, it can be relaxed into a standard SDP dual

$$\begin{aligned} \min_{M,\boldsymbol{\mu},\boldsymbol{\nu},t} \quad & t \\ \text{s.t.} \quad & M \succeq \mathbf{0} \\ & \operatorname{diag}(M) = \mathbf{1}_n \\ & \boldsymbol{\mu} \geq \mathbf{0}_n, \boldsymbol{\nu} \geq \mathbf{0}_n \\ & \begin{pmatrix} M \circ K & (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu}) \\ (\mathbf{1}_n - \boldsymbol{\mu} + \boldsymbol{\nu})^\top & t - 2C\boldsymbol{\mu}^\top \mathbf{1}_n \end{pmatrix} \succeq \mathbf{0}, \end{aligned} \tag{22}$$

and solved by any standard SDP solver in $O(n^{6.5})$ time.

Remark 21 Xu et al. (2005) initially imposed three groups of linear constraints on the entries of M in MMC:

1. $\forall ijk$, $M_{i,k} \geq M_{i,j} + M_{j,k} - 1$;
2. $\forall ijk$, $M_{i,k} \geq -M_{i,j} - M_{j,k} - 1$;
3. $\forall i$, $-b \leq \sum_j M_{i,j} \leq b$.

However, Xu and Schuurmans (2005) and Valizadegan and Jin (2007) considered (22) as the dual problem of MMC, sometimes equipped with an additional class balance constraint $-b\mathbf{1}_n \leq M\mathbf{1}_n \leq b\mathbf{1}_n$. In other words, the first and second groups of constraints were ignored.

Subsequently, *generalized maximum margin clustering* (GMMC) (Valizadegan and Jin, 2007) relaxes the restriction that the original MMC only considers homogeneous hyperplanes and hence demands every possible clustering boundary to pass through the origin. Furthermore, GMMC is a convex relaxation of MMC, and its computational complexity is $O(n^{4.5})$ that is remarkably faster than MMC. In fact, GMMC optimizes an n-dimensional vector rather than an $n \times n$ matrix. More specifically, the hard-margin GMMC converts the original MMC following Lanckriet et al. (2004) into a dual problem as

$$\begin{aligned} \min_{\boldsymbol{y} \in \{-1,+1\}^n} \min_{\boldsymbol{\nu},\lambda} \quad & \frac{1}{2}(\mathbf{1}_n + \boldsymbol{\nu} + \lambda \boldsymbol{y})^\top \operatorname{diag}(\boldsymbol{y}) K^{-1} \operatorname{diag}(\boldsymbol{y})(\mathbf{1}_n + \boldsymbol{\nu} + \lambda \boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{\nu} \geq \mathbf{0}_n, \end{aligned}$$

where the function $\operatorname{diag}(\cdot)$ here converts a column vector into a diagonal matrix. The trick here is

$$\left(K \circ \boldsymbol{y}\boldsymbol{y}^\top\right)^{-1} = (\operatorname{diag}(\boldsymbol{y}) K \operatorname{diag}(\boldsymbol{y}))^{-1} = \operatorname{diag}(\boldsymbol{y}) K^{-1} \operatorname{diag}(\boldsymbol{y}),$$

since $\boldsymbol{y} \in \{-1,+1\}^n$. By a tricky substitution $\boldsymbol{w} = (\operatorname{diag}(\boldsymbol{y})(\mathbf{1}_n + \boldsymbol{\nu}); \lambda) \in \mathbb{R}^{n+1}$ where we use the semicolon to separate the rows of a vector or matrix (i.e., $(A;B) = (A^\top, B^\top)^\top$), it becomes

$$\begin{aligned} \min_{\boldsymbol{w} \in \mathbb{R}^{n+1}} \quad & \boldsymbol{w}^\top (I_n; \mathbf{1}_n^\top) K^{-1} (I_n, \mathbf{1}_n) \boldsymbol{w} + C_e \left((\mathbf{1}_n^\top, 0) \boldsymbol{w} \right)^2 \\ \text{s.t.} \quad & [\boldsymbol{w}]_i^2 \geq 1, i = 1, \ldots, n, \end{aligned} \tag{23}$$

where $((\mathbf{1}_n^\top, 0)\boldsymbol{w})^2$ is another regularization to remove the translation invariance from the objective function and C_e is the corresponding regularization parameter. Let

$$W = (I_n; \mathbf{1}_n^\top) K^{-1} (I_n, \mathbf{1}_n) + C_e (\mathbf{1}_n; 0)(\mathbf{1}_n^\top, 0) - \mathrm{diag}((\boldsymbol{\gamma}; 0)).$$

The SDP dual of optimization (23) is then

$$\max_{\boldsymbol{\gamma} \in \mathbb{R}^n} \boldsymbol{\gamma}^\top \mathbf{1}_n \qquad \text{s.t. } W \succeq \mathbf{0}, \boldsymbol{\gamma} \geq \mathbf{0}_n.$$

This is the dual problem of the hard-margin GMMC. The dual problem of the soft-margin GMMC is slightly different such that $\boldsymbol{\gamma}$ is upper bounded:

$$\max_{\boldsymbol{\gamma} \in \mathbb{R}^n} \boldsymbol{\gamma}^\top \mathbf{1}_n \qquad \text{s.t. } W \succeq \mathbf{0}, \mathbf{0}_n \leq \boldsymbol{\gamma} \leq C_\delta \mathbf{1}_n, \tag{24}$$

where C_δ is a regularization parameter to control the trade-off between the clustering error and the margin. After obtaining the optimal $\boldsymbol{\gamma}$, the partition can be inferred from the sign of the eigenvector of W associated with the zero eigenvalue, since the Karush-Kuhn-Tucker complementary condition is $W\boldsymbol{w} = \mathbf{0}_{n+1}$, and $\mathrm{sign}([\boldsymbol{w}]_i) = \mathrm{sign}([\boldsymbol{y}]_i)$ for $i = 1, \ldots, n$.

There exist a few faster MMC algorithms. *Iterative support vector regression* (IterSVR) (Zhang et al., 2007) replaces SVM with the hinge loss in the inner optimization subproblem with SVR with the Laplacian loss, while for each inner SVR the time complexity is at most $O(n^3)$ and the empirical time complexity is usually between $O(n)$ and $O(n^{2.3})$. *Cutting-plane maximum margin clustering* (CPMMC) (Zhao et al., 2008b) can be solved by a series of constrained concave-convex procedures within a linear time complexity $O(sn)$ where s is the average number of non-zero features. Unlike MMC and GMMC that rely on SDP or IterSVR and CPMMC that are non-convex, *label-generation maximum margin clustering* (LGMMC) (Li et al., 2009) is scalable yet convex so that it can achieve its globally optimal solution. Roughly speaking, LGMMC replaces the hinge loss in SVM with the squared hinge loss to get an alternative MMC:

$$\begin{aligned} \min_{\boldsymbol{y} \in \{-1,+1\}^n} \min_{\boldsymbol{w}, \boldsymbol{\xi}} \; & \frac{1}{2}\|\boldsymbol{w}\|_2^2 - \rho + \frac{C}{2}\sum_{i=1}^n \xi_i^2 \\ \text{s.t. } & y_i \boldsymbol{w}^\top x_i \geq \rho - \xi_i, \quad i = 1, \ldots, n \\ & -b \leq \boldsymbol{y}^\top \mathbf{1}_n \leq b. \end{aligned}$$

After a long derivation, LGMMC can be expressed as a *multiple kernel learning* problem:

$$\begin{aligned} \min_{\boldsymbol{\mu} \in \mathbb{R}^{2^n}} \max_{\boldsymbol{\alpha}} \; & -\frac{1}{2}\boldsymbol{\alpha}^\top \left(\sum\nolimits_{t: -b \leq \boldsymbol{y}_t^\top \mathbf{1}_n \leq b} \mu_t K \circ \boldsymbol{y}_t \boldsymbol{y}_t^\top + \frac{1}{C} I_n \right) \boldsymbol{\alpha} \\ \text{s.t. } & \boldsymbol{\mu}^\top \mathbf{1}_{2^n} = 1, \boldsymbol{\mu} \geq \mathbf{0}_{2^n} \\ & \boldsymbol{\alpha}^\top \mathbf{1}_n = 1, \boldsymbol{\alpha} \geq \mathbf{0}_n. \end{aligned}$$

This optimization is again solved by the cutting plane method, that is, finding the most violated $\boldsymbol{y}_t$ iteratively, and the empirical time complexity of multiple kernel learning has the same order as the complexity of SVM which usually scales between $O(n)$ and $O(n^{2.3})$.

On the other hand, the stability of MVC is by no means inferior to those non-convex MMC in terms of the resulting clusters. The optimization involved in MVC-HL is a convex SDP problem;

the optimization involved in MVC-SL is a non-convex SQP problem, while under mild conditions, it seems convex if one only cares the resulting clusters. Moreover, MVC-SL has a clustering error bound, and to the best of our knowledge no MMC has such a result. Although the asymptotic time complexity of MVC-SL is $O(n^3)$, its computation time has exhibited less potential of growth in our experiments than the computationally-efficient LGMMC (see Figure 5 in page 2669).

7.2 Spectral Clustering

Spectral clustering (SC) (Shi and Malik, 2000; Meila and Shi, 2001; Ng et al., 2002) is also closely related to MVC. SC algorithms include two steps, a spectral embedding step to unfold the manifold structure and embed the input data into a low-dimensional space in a geodesic manner, and then a k-means clustering step to carry out clustering using the embedded data.

Given a similarity matrix $W \in \mathbb{R}^{n\times n}$ and the corresponding degree matrix $D = \mathrm{diag}(W\mathbf{1}_n)$, we have three popular graph Laplacian matrices: The unnormalized graph Laplacian is defined as

$$L_{\mathrm{un}} := D - W,$$

and two normalized graph Laplacian are

$$L_{\mathrm{sym}} := D^{-1/2} L_{\mathrm{un}} D^{-1/2} = I_n - D^{-1/2} W D^{-1/2}$$
$$L_{\mathrm{rw}} := D^{-1} L_{\mathrm{un}} = I_n - D^{-1} W.$$

The first matrix is denoted by L_{sym} since it is a symmetric matrix and the second one by L_{rw} since it is closely related to a random walk. Each popular graph Laplacian corresponds to a popular SC algorithm according to von Luxburg (2007). Unnormalized SC computes the first k eigenvectors of L_{un} where the eigenvalues are all positive and listed in an increasing order. Shi and Malik (2000) computes the first k generalized eigenvectors of the generalized eigenvalue problem $L_{\mathrm{un}}\boldsymbol{u} = \lambda D\boldsymbol{u}$ that are also the eigenvectors of L_{rw}, and hence it is called normalized SC.[9] The other normalized SC algorithm, namely Ng et al. (2002), computes the first k eigenvectors of L_{sym}, puts them into an $n \times k$ matrix, and normalizes all rows of that matrix to the unit norm, that is, projects the embedded data further to the k-dimensional unit sphere. Anyway, the main idea is to change the representation from $\mathbb{R}^d$ to $\mathbb{R}^k$ and then run k-means clustering.

MVC-SL is able to integrate the two steps of unnormalized SC into a single optimization when the number of clusters is two and the highly non-convex k-means step is unnecessary. Furthermore, a vital difference between MVC and SC is that the basic model of MVC has a loss function which pushes hypotheses away from the coordinate axes and always leads to non-sparse optimal solutions. When considering the finite sample stability, the spectral embedding step of SC is stable if MVC-SL is stable but not vice versa, since SC only requires that the graph Laplacian has distinct eigenvalues; the k-means step is always unstable for fixed data due to the non-convex distortion function which is essentially an integer programming, but it is stable for different random samplings from the same underlying distribution, if the globally optimal solution is unique (Rakhlin and Caponnetto, 2007). In addition, there are a few theoretical results about the infinite sample stability or the consistency of SC. Globally optimal solutions to k-means clustering converge to a limit partition of the whole data space $\mathcal{X}$, if the underlying distribution has a finite support, and the globally optimal solution

9. Actually, two algorithms were proposed in Shi and Malik (2000): The two-way cut algorithm only makes use of the second eigenvector and the k-way cut algorithm uses all first k eigenvectors.

to the expectation of the distortion function with respect to the underlying distribution is unique (Ben-David et al., 2007). Eigenvectors of graph Laplacian also converge to eigenvectors of certain limit operators, while the conditions for convergence are very general for L_{sym}, but are very special for L_{un} so that they are not easily satisfied (von Luxburg et al., 2005, 2008). In contrast, the infinite sample stability of MVC is currently an open problem.

Remark 22 Certain SC algorithms such as Belkin and Niyogi (2002) ignore the first eigenvector by extracting the second to k-th eigenvectors of some graph Laplacian, and thus change the representation to $\mathbb{R}^{k-1}$ rather than $\mathbb{R}^k$. Nevertheless, the multiplicity of the eigenvalue zero of the graph Laplacian equals the number of connected components of the similarity graph, and the eigenspace of eigenvalue zero is spanned by the indicator vectors of the connected components (von Luxburg, 2007, Propositions 2 and 4). As a consequence, all three aforementioned SC algorithms keep the first eigenvector in order to deal with disconnected similarity graphs.

7.3 Approximate Volume Regularization

The connection of *approximate volume regularization* (AVR) (El-Yaniv et al., 2008) and MVC is analogous with the connection of SVM and MMC.

Compared with MVC, AVR is a transductive method for classification so that the label vector $\boldsymbol{y}$ is constant and only the soft response vector $\boldsymbol{h}$ needs to be optimized. More specifically, given m labeled data $\{(x_1,y_1),\ldots,(x_m,y_m)\}$ and u unlabeled data $\{x_{m+1},\ldots,x_{m+u}\}$, the label vector is denoted by $\boldsymbol{y}=(y_1,\ldots,y_m,0,\ldots,0)^\top\in\mathbb{R}^{m+u}$, and the primal problem of AVR is defined as

$$\min_{\boldsymbol{h}\in\mathbb{R}^{m+u}} -\frac{1}{m}\boldsymbol{h}^\top\boldsymbol{y}+\gamma\boldsymbol{h}^\top Q\boldsymbol{h} \qquad \text{s.t. } \|\boldsymbol{h}\|_2=t, \tag{25}$$

where t is a hyperparameter to control the scale of $\boldsymbol{h}$. Since $\boldsymbol{y}$ is constant, optimization (25) can be directly solved using Lagrangian multipliers and the Karush-Kuhn-Tucker conditions

$$\begin{aligned} -\boldsymbol{y}/m+2\gamma Q\boldsymbol{h}-2\eta\boldsymbol{h}&=0,\\ \boldsymbol{h}^\top\boldsymbol{h}-t^2&=0. \end{aligned}$$

Let the eigen-decomposition of Q be $Q=V\Lambda V^\top$ and $d_i=[V^\top\boldsymbol{y}]_i$, then we get an equation about the optimal η:

$$\frac{1}{4m^2}\sum_{i=1}^{m+u}\frac{d_i^2}{(\gamma\lambda_i-\eta)^2}-t^2=0. \tag{26}$$

Thanks to the special structure of (26), a binary search procedure is enough for finding its smallest root η^*, and the optimal $\boldsymbol{h}$ is recovered by

$$\boldsymbol{h}^*=\frac{1}{2m}(\gamma Q-\eta^* I_{m+u})^{-1}\boldsymbol{y}.$$

On the other hand, MVC involves a combinatorial optimization similarly to the most clustering models and several semi-supervised learning models such as MMC. This difficulty caused by the integer feasible region is intrinsically owing to the clustering problem and has no business with the large volume approximation $V(\boldsymbol{h})$. In order to solve the basic model, we proposed two approximation schemes based on sequential quadratic programming and semi-definite programming that are much more complicated than finding the smallest root of Equation (26) as in AVR.

8. Experiments

In this section, we numerically evaluate the performance of the proposed MVC algorithms.

8.1 Setup

Seven clustering algorithms were included in our experiments:

- Kernel k-means clustering (KM; Zha et al., 2002),
- Normalized spectral clustering (NSC; Ng et al., 2002),
- Maximum margin clustering (MMC; Xu et al., 2005),
- Generalized MMC (GMMC; Valizadegan and Jin, 2007),
- Label-generation MMC (LGMMC; Li et al., 2009),
- Soft-label maximum volume clustering (MVC-SL),
- Hard-label maximum volume clustering (MVC-HL).

The *CVX* package (Grant and Boyd, 2011), which is a Matlab-based modeling system for disciplined convex programming, was used to solve the QP problem (5) for MVC-SL and the SDP problems (13), (22) and (24) for MVC-HL, MMC and GMMC.

Table 1 summarizes the specification of data sets in our experiments. We first evaluated all seven algorithms on three artificial data sets. MVC-HL and MMC were excluded from the middle-scale experiments since they were very time-consuming when $n > 100$. The *IDA benchmark repository*[10] contains thirteen benchmark data sets for binary classification, and ten of them that have no intrinsic within-class multi-modality were included. Additionally, we made intensive comparisons based on four well-known benchmark data sets for classification: *USPS* and *MNIST*[11] contain 8-bit gray-scale images of handwritten digits '0' through '9' with the resolution 16×16 and 28×28, *20Newsgroups sorted by date*[12] contains term-frequency vectors of documents that come from twenty newsgroups, and *Isolet*[13] contains acoustic features of isolated spoken letters from 'A' to 'Z'.

In our experiments, the performance was measured by the clustering error rate

$$\frac{1}{n} d_{\mathcal{H}}(\boldsymbol{y}, \boldsymbol{y}^*) = \frac{1}{2n} \min(\|\boldsymbol{y} + \boldsymbol{y}^*\|_1, \|\boldsymbol{y} - \boldsymbol{y}^*\|_1),$$

where $\boldsymbol{y}$ is the label vector returned by clustering algorithms and $\boldsymbol{y}^*$ is the ground truth label vector. The similarity measure was either the Gaussian similarity

$$W_{i,j} = \exp\left(-\frac{\|x_i - x_j\|_2^2}{2\sigma^2}\right)$$

with a hyperparameter σ, the cosine similarity

$$W_{i,j} = \begin{cases} \dfrac{\langle x_i, x_j \rangle}{\|x_i\|_2 \|x_j\|_2} & \text{if } x_i \sim_k x_j, \\ 0 & \text{otherwise,} \end{cases}$$

10. The data sets were downloaded from `http://ida.first.gmd.de/~raetsch/data/benchmarks.htm`.
11. The data sets are available at `http://cs.nyu.edu/~roweis/data.html`.
12. The data set is available at `http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html`.
13. The data set is available at `http://archive.ics.uci.edu/ml/datasets/isolet`.

	# Classes	# Features	# Data	# Samplings
Artificial Data				
2gaussians	2	3	-	12×10
2moons	2	2	400	12×10
2circles	2	2	315	12×10
IDA Benchmarks				
Breast-cancer	2	9	200	100
Diabetes	2	8	468	100
Flare-solar	2	9	666	100
German	2	20	700	100
Heart	2	13	170	100
Image	2	18	1300	20
Ringnorm	2	20	400	100
Splice	2	60	1000	20
Titanic	2	3	150	100
Twonorm	2	20	400	100
Other Benchmarks				
USPS	10	256	11000	8×10
MNIST	10	784	70000	8×10
20Newsgroups	7	26214	18846	8×10
Isolet	26	617	7797	8×10

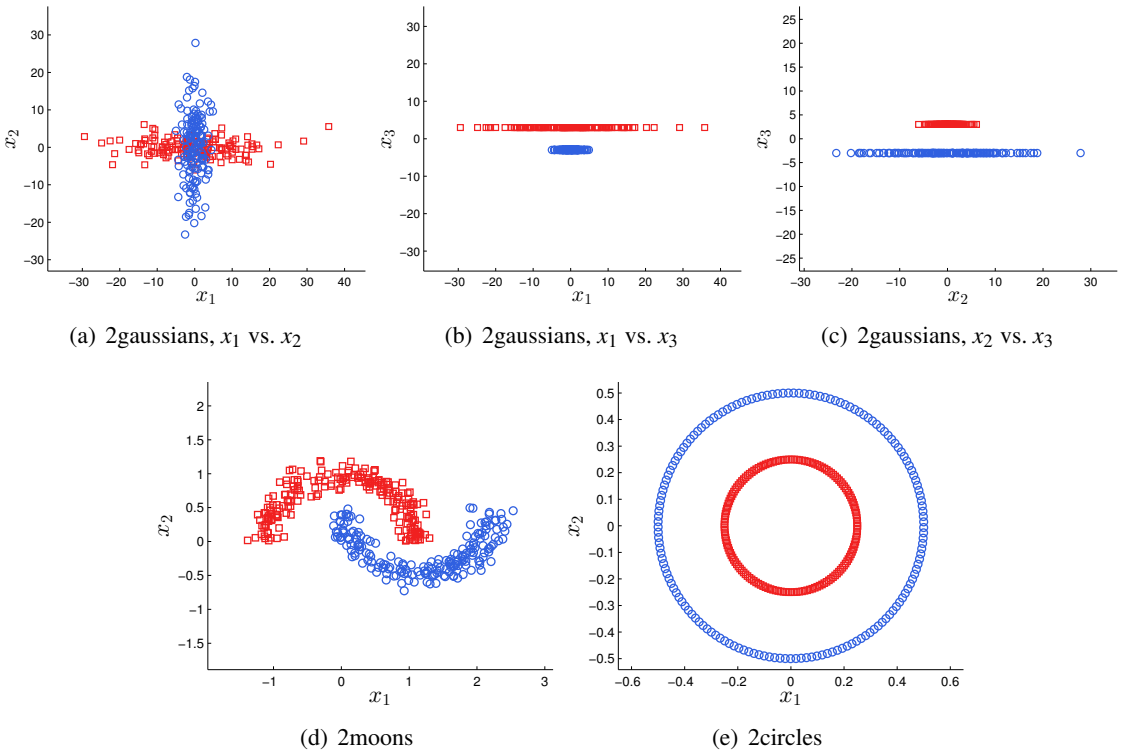
Table 1: Specification of artificial and benchmark data sets.

with a hyperparameter k, where $x_i \sim_k x_j$ means that x_i and x_j are among the k-nearest neighbors of each other, or the locally-scaled Gaussian-like similarity (Zelnik-Manor and Perona, 2005)

$$W_{i,j} = \exp\left(-\frac{\|x_i - x_j\|_2^2}{2\sigma_i\sigma_j}\right)$$

with a hyperparameter k, where $\sigma_i = \|x_i - x_i^{(k)}\|_2$ is called the local scaling factor of x_i and $x_i^{(k)}$ is the k-th nearest neighbor of x_i in X_n. The kernel matrix was $K = W$ for KM, MMC and LGMMC, and $K = W + I_n/n$ for GMMC since it would be very unstable without this small eigenvalue shift. NSC relied on the graph Laplacian L_{sym} constructed from W. Due to the requirement of positive definiteness of Q for MVC, we also slightly shifted the eigenvalues of certain positive semi-definite matrices and adopted $Q = L_{\mathrm{sym}} + I_n/n$ for MVC-SL and $Q = W + I_n/n$ for MVC-HL.

Numerical issues always exist and there may be more than one candidate $\boldsymbol{h}_0$ for MVC-SL. Let $\lambda_1 \leq \cdots \leq \lambda_n$ be the eigenvalues of Q, and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the associated normalized eigenvectors. In our implementation, we initialize MVC-SL by a few eigenvectors whose eigenvalues are close to λ_2. Specifically, we construct a set of candidate eigenvectors $\mathcal{V} = \{\boldsymbol{v}_i \mid |\lambda_i - \lambda_2| < 10^{-4}\}$, and if $\#\mathcal{V} > 10$, we say that Q is ill-defined and only keep ten such $\boldsymbol{v}_i$ in $\mathcal{V}$. Next we obtain one $\boldsymbol{h}_0$ from each $\boldsymbol{v}_i \in \mathcal{V}$ and solve the SQP problem based on each $\boldsymbol{h}_0$. At last, the solution $\boldsymbol{h}^*$ resulting in the smallest objective value $-2\|\boldsymbol{h}^*\|_1 + \gamma \boldsymbol{h}^{*\top} Q \boldsymbol{h}^*$ would be selected as the final solution to MVC-SL. This trick can sometimes improve the performance significantly, while the cost is the increase of the computation time by no more than ten times.



(a) 2gaussians, x_1 vs. x_2 (b) 2gaussians, x_1 vs. x_3 (c) 2gaussians, x_2 vs. x_3

(d) 2moons (e) 2circles

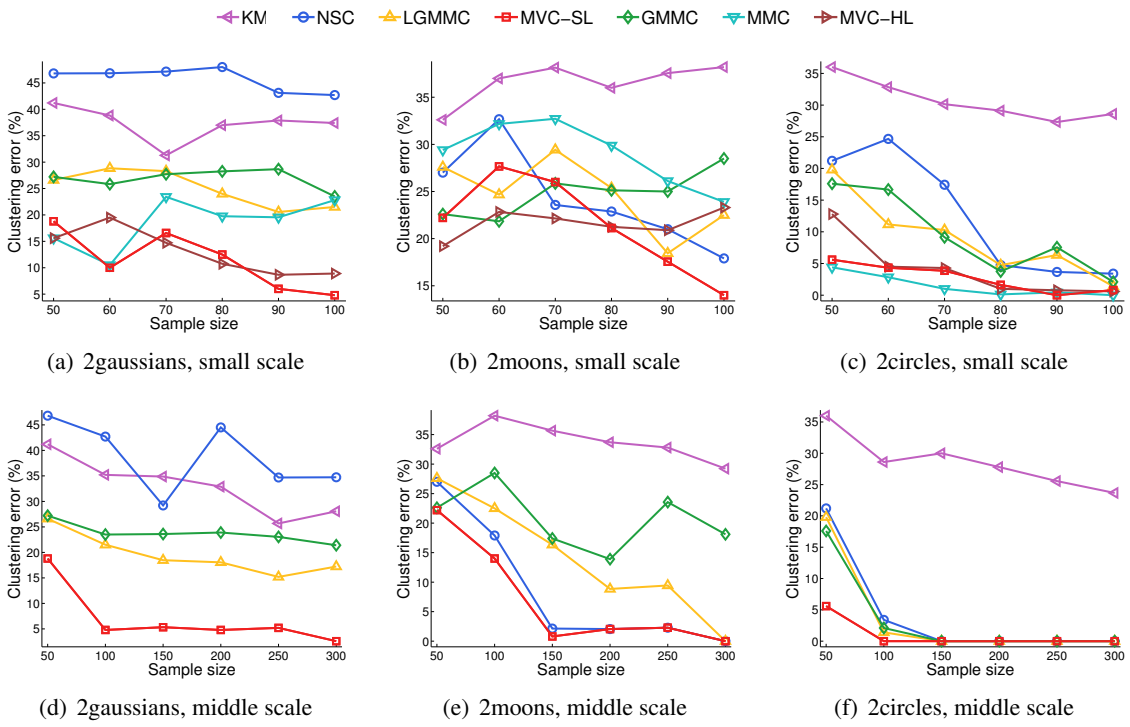
Figure 3: Visualization of artificial data sets.

8.2 Artificial Data Sets

To begin with, we compare the clustering error and the computation time of all seven algorithms based on three artificial data sets. As visualized in Figure 3, *2gaussians* is a three-dimensional data set generated as follows. We first randomly sampled $X_{n/2}^+$ from a Gaussian distribution with zero mean and covariance matrix $\mathrm{diag}(100,4)$ and $X_{n/2}^-$ from the other Gaussian distribution with zero mean and covariance matrix $\mathrm{diag}(4,100)$, set the third dimension as $+3$ for $X_{n/2}^+$ and -3 for $X_{n/2}^-$ and combined $X_{n/2}^+$ and $X_{n/2}^-$ into X_n. Subsequently, *2moons* is a two-dimensional data set with two non-Gaussian crescent-like clusters, and *2circles* is another two-dimensional data set with two non-Gaussian ring-like clusters. The Gaussian similarity was applied to all algorithms, and σ was fixed to $m_\sigma/10$, where m_σ is the mean pairwise distance given by

$$m_\sigma = \frac{\sum_{1\le i<j\le n}\|x_i-x_j\|_2}{n(n-1)/2} = \frac{\sum_{i,j=1}^n\|x_i-x_j\|_2}{n(n-1)}, \tag{27}$$

since $\|x_i-x_j\|_2=0$ when $i=j$. Then, the regularization parameter C of MMC was the best value among $\{10^{-3},1,10^3\}$, that is, we ran MMC three times using $C=10^{-3},1,10^3$ and recorded the best performance, since there lacks a uniformly effective model selection framework for clustering algorithms. The regularization parameter C of LGMMC was also selected from $\{10^{-3},1,10^3\}$ in the same way. For GMMC, the regularization parameter C_e was set to 10^4 following Valizadegan and Jin (2007) and the other regularization parameter C_δ was the best candidate in $\{10^{-3},1,10^3\}$. We fixed the stopping threshold ε to 10^{-6}, the regularization parameter γ to 10^{-2} and let the class



(a) 2gaussians, small scale (b) 2moons, small scale (c) 2circles, small scale

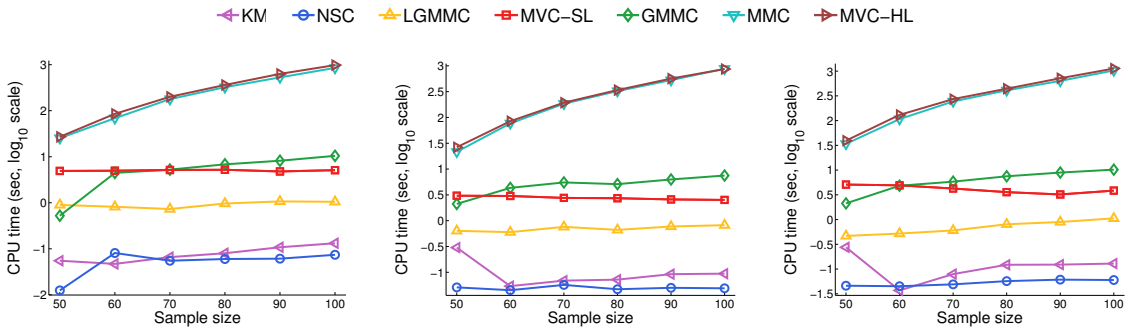
(d) 2gaussians, middle scale (e) 2moons, middle scale (f) 2circles, middle scale

Figure 4: Means of the clustering error (in %) on 2gaussians, 2moons and 2circles.

balance parameter b adaptively be $1/n$ for MVC-SL, while for MVC-HL, we fixed C to 1 and tried $\gamma \in \{10^{-3}, 1, 10^3\}$.

The experimental results in terms of the means of the clustering error are reported in Figure 4. All of the results were obtained by repeatedly running an algorithm on 10 random samplings with given sample size n, and the sample sizes were $\{50, 60, 70, 80, 90, 100\}$ for the small-scale experiments and $\{50, 100, 150, 200, 250, 300\}$ for the middle-scale experiments. We can see that among the three data sets, 2gaussians is most difficult such that LGMMC still had a mean clustering error around twenty percents even when $n = 300$, and 2circles is easiest because MMC, MVC-SL and MVC-HL already got near zero errors when $n = 80$ and LGMMC, GMMC and NSC also achieved perfect partitions after $n = 150$. In contrast, KM cannot deal with these artificial data well due to the non-convex distortion function and the random initialization of cluster centers, even though it was equipped with the Gaussian similarity. Surprisingly, NSC was worse than KM on 2guassians, whereas MVC-SL based on the almost same input $Q = L_{\mathrm{sym}} + I_n/n$ had much lower clustering errors, which implies that the highly non-convex k-means step may be a bottleneck of NSC.

Next we report the corresponding computation time of these algorithms in Figure 5. All of the results were measured in average seconds per run on Xeon X5670 processors. Note that the worst case running time (i.e., the asymptotic time complexity) of KM is super-polynomial in the sample size n (Arthur and Vassilvitskii, 2006), and so is the worst case running time of NSC. On the other hand, the asymptotic time complexities of LGMMC, MVC-SL, GMMC, MMC and MVC-HL are $O(n^3)$, $O(n^3)$, $O(n^{4.5})$, $O(n^{6.5})$ and $O(n^{6.5})$, respectively. In our experiments, NSC was the most computationally-efficient algorithm and almost always faster than KM, since the k-means invoked by NSC after the spectral embedding converged in fewer iterations than KM. While LGMMC was



(a) 2gaussians, small scale (b) 2moons, small scale (c) 2circles, small scale

(d) 2gaussians, middle scale (e) 2moons, middle scale (f) 2circles, middle scale

Figure 5: Means of the CPU time (in sec, per run) on 2gaussians, 2moons and 2circles.

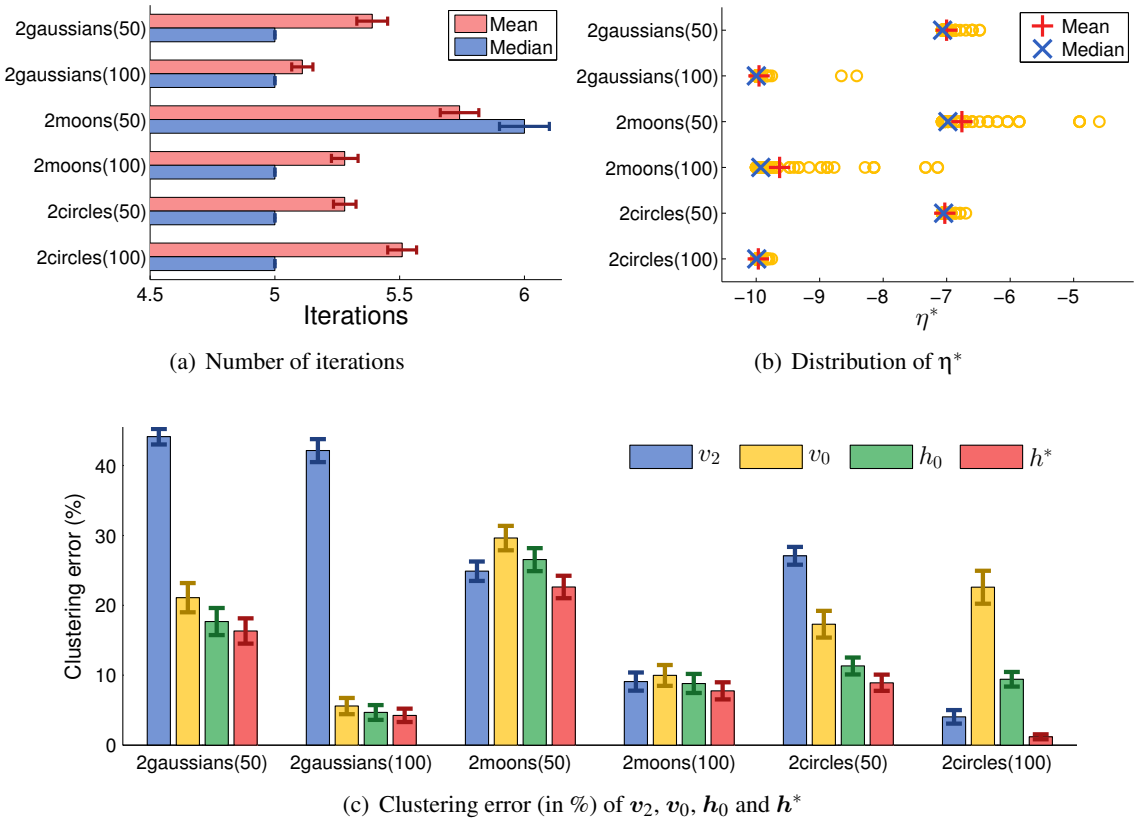
consistently faster than GMMC, MVC-SL lay between them and was comparable with GMMC in the small-scale experiments and comparable with LGMMC in the middle-scale experiments. As a result, the computation time or empirical time complexity of MVC-SL exhibited less potential of growth than LGMMC and GMMC. The worst-case computational complexities of MVC-HL and MMC made them extremely time-consuming, poorly scalable to middle or large sample sizes, and hence impractical despite their low mean clustering errors on 2guassians and 2circles.

Furthermore, we investigate three important properties of MVC-SL, and report the results over 100 random samplings in Figure 6.

Firstly, panel (a) shows the mean and median values about the number of iterations required by MVC-SL, where each mean is shown with the *standard error*, and each median is shown with the *median absolute deviation* divided by the square root of the number of random samplings (i.e., 10). As mentioned before, the convergence rate of SQP iterations is independent of the sample size n, and we can see that MVC-SL usually stopped within just a few iterations in our experiments. This phenomenon implies that the empirical time complexity of MVC-SL is directly proportional to the internal QP solver.

Secondly, we examine the distribution of η^* which may influence the stability of the resulting clusters. Fortunately, panel (b) shows that η^* for fixed data set and fixed sample size were highly concentrated, and the mean and median values exhibited a strong correlation with the sample size as well as a weak correlation with the data set.

Thirdly, recall that there may be more than one candidate $\boldsymbol{h}_0$ and we initialize MVC-SL using $\mathcal{V} = \{\boldsymbol{v}_i \mid |\lambda_i - \lambda_2| < 10^{-4}\}$. Although all $\boldsymbol{v} \in \mathcal{V}$ appear nearly equally good to NSC, they could induce initial solutions of very different qualities for MVC-SL, as shown in panel (c). The vectors



(a) Number of iterations

(b) Distribution of η^*

(c) Clustering error (in %) of $\boldsymbol{v}_2$, $\boldsymbol{v}_0$, $\boldsymbol{h}_0$ and $\boldsymbol{h}^*$

Figure 6: Experimental results concerning three important properties of MVC-SL.

$\boldsymbol{v}_2$, $\boldsymbol{v}_0$, $\boldsymbol{h}_0$, and $\boldsymbol{h}^*$ are all treated as soft response vectors, and the means with standard errors of the clustering error are plotted in panel (c), where $\boldsymbol{v}_2$ is the eigenvector of Q and L_{sym} associated with λ_2, $\boldsymbol{v}_0$ is the eigenvector selected by MVC-SL, and $\boldsymbol{h}_0$ and $\boldsymbol{h}^*$ are the corresponding initial and final solutions. We can see that $\boldsymbol{h}^*$ was better than $\boldsymbol{h}_0$ and $\boldsymbol{h}_0$ was better than $\boldsymbol{v}_0$. Moreover, $\boldsymbol{v}_0$ was significantly superior to $\boldsymbol{v}_2$ on 2guassians. It is interesting and surprising that both $\boldsymbol{h}_0$ and $\boldsymbol{v}_0$ were significantly inferior to $\boldsymbol{v}_2$ on 2circles when $n = 100$, but they still resulted in $\boldsymbol{h}^*$ with the lowest mean clustering error. In a word, not only good initial solutions but also the SQP method contribute to the success of MVC-SL, which in turn implies that the underlying large volume principle should be reasonable for clustering.

8.3 Benchmark Data Sets

In the following, we discuss the experiments on the benchmarks listed in Table 1: The experiments involving ten IDA benchmarks are discussed in the first part, then USPS and MNIST in the second part, 20Newsgroups in the third part, and Isolet in the fourth part.

8.3.1 IDA BENCHMARKS

We compare KM, NSC, LGMMC, GMMC, and MVC-SL on ten data sets in the IDA benchmark repository that are designed for binary classification tasks and have one hundred fixed realizations

	KM	NSC	LGMMC	MVC-SL	GMMC	SVM
Breast-cancer	38.9 ± 0.65	26.4 ± 0.18	27.2 ± 0.19	$\mathbf{25.6 \pm 0.17}$	30.5 ± 0.23	26.0
Diabetes	30.3 ± 0.17	30.6 ± 0.18	$\mathbf{27.6 \pm 0.13}$	30.4 ± 0.15	28.6 ± 0.15	23.5
Flare-solar	$\mathbf{35.5 \pm 0.20}$	44.9 ± 0.11	37.6 ± 0.16	44.5 ± 0.12	N/A	32.4
German	39.4 ± 0.20	$\mathbf{30.2 \pm 0.09}$	$\mathbf{30.1 \pm 0.09}$	$\mathbf{30.2 \pm 0.09}$	N/A	23.6
Heart	$\mathbf{18.5 \pm 0.38}$	$\mathbf{18.0 \pm 0.23}$	18.7 ± 0.28	18.8 ± 0.22	18.9 ± 0.21	16.0
Image	41.0 ± 0.36	40.5 ± 0.15	$\mathbf{39.7 \pm 0.20}$	40.9 ± 0.11	N/A	2.96
Ringnorm	4.68 ± 0.11	$\mathbf{2.20 \pm 0.06}$	6.61 ± 0.11	$\mathbf{2.17 \pm 0.06}$	$\mathbf{2.07 \pm 0.06}$	1.66
Splice	29.1 ± 1.41	35.5 ± 0.44	$\mathbf{25.5 \pm 0.72}$	36.1 ± 0.44	N/A	10.9
Titanic	27.2 ± 0.59	26.8 ± 0.42	23.1 ± 0.36	$\mathbf{21.9 \pm 0.37}$	26.1 ± 0.43	22.4
Twonorm	3.61 ± 0.78	2.28 ± 0.07	$\mathbf{2.18 \pm 0.07}$	$\mathbf{2.20 \pm 0.07}$	$\mathbf{2.08 \pm 0.06}$	2.96

Table 2: Means with standard errors of the clustering error (in %) on IDA benchmark data sets. For each data set, the best algorithm and comparable ones based on the unpaired t-test at the significance level 5% are highlighted in boldface. Additionally, means of the classification error of highly-tuned SVM provided by IDA are also listed for comparison.

for each data set except that the data sets Image and Splice only have twenty realizations. For each realization of each data set, we ignored the test data and tested five clustering algorithms using the training data, yet GMMC was not tested on the data sets Flare-solar, German, Image and Splice as it required a very long execution time when $n \geq 600$. The Gaussian similarity was applied and σ was the best value among $\{4m_\sigma, 2m_\sigma, m_\sigma, m_\sigma/2, m_\sigma/4\}$ for each realization and each algorithm, where the variable m_σ was the mean pairwise distance defined in Equation (27). An exception is the data set Ringnorm where the locally-scaled similarity with $k = 7$ was applied, since it consists of data from two highly overlapped Gaussian distributions and can be treated as a multi-scale data set.[14] The settings for other hyperparameters of LGMMC, GMMC, and MVC-SL were exactly same as the experiments on the artificial data sets, specifically, $C \in \{10^{-3}, 1, 10^3\}$ for LGMMC, $C_e = 10^4$ and $C_\delta \in \{10^{-3}, 1, 10^3\}$ for GMMC, and $\varepsilon = 10^{-6}$, $\gamma = 10^{-2}$ and $b = 1/n$ for MVC-SL.

Table 2 describes the means with standard errors of the clustering error rate by each algorithm on each data set. For the sake of comparison, Table 2 also lists the means of the classification error rate of highly-tuned SVM provided by the official web site of the IDA benchmark repository.

We could see from Table 2 that LGMMC and MVC-SL were either the best algorithm or comparable to the best algorithm based on the unpaired t-test at the significance level 5% on five data sets. The clustering errors of five algorithms exhibited large differences on five data sets, namely, Breast-cancer, Flare-solar, German, Ringnorm and Splice, among which MVC-SL was one of the best algorithms on three data sets, and LGMMC was one of the best algorithms on two data sets. The clustering errors exhibited merely small differences on the other five data sets. Moreover, the fully supervised SVM has a mean classification error obviously smaller than the lowest mean clustering error on the data sets German, Image and Splice, and larger than the lowest mean clustering error on the data sets Breast-cancer, Titanic and Twonorm. It should not be surprising or confusing since the classification error is the out-of-sample test error on the test data whereas the clustering error is the in-sample test error on the same data to be clustered.

14. In fact, Ringnorm violates the underlying assumption when evaluating clustering results using classification data sets, that is, the class structure and the cluster structure must coincide with each other. However, 2circles does not violate this assumption, since those ring-like clusters are neither Gaussian distributions nor overlapped clusters.

8.3.2 IMAGES OF HANDWRITTEN DIGITS

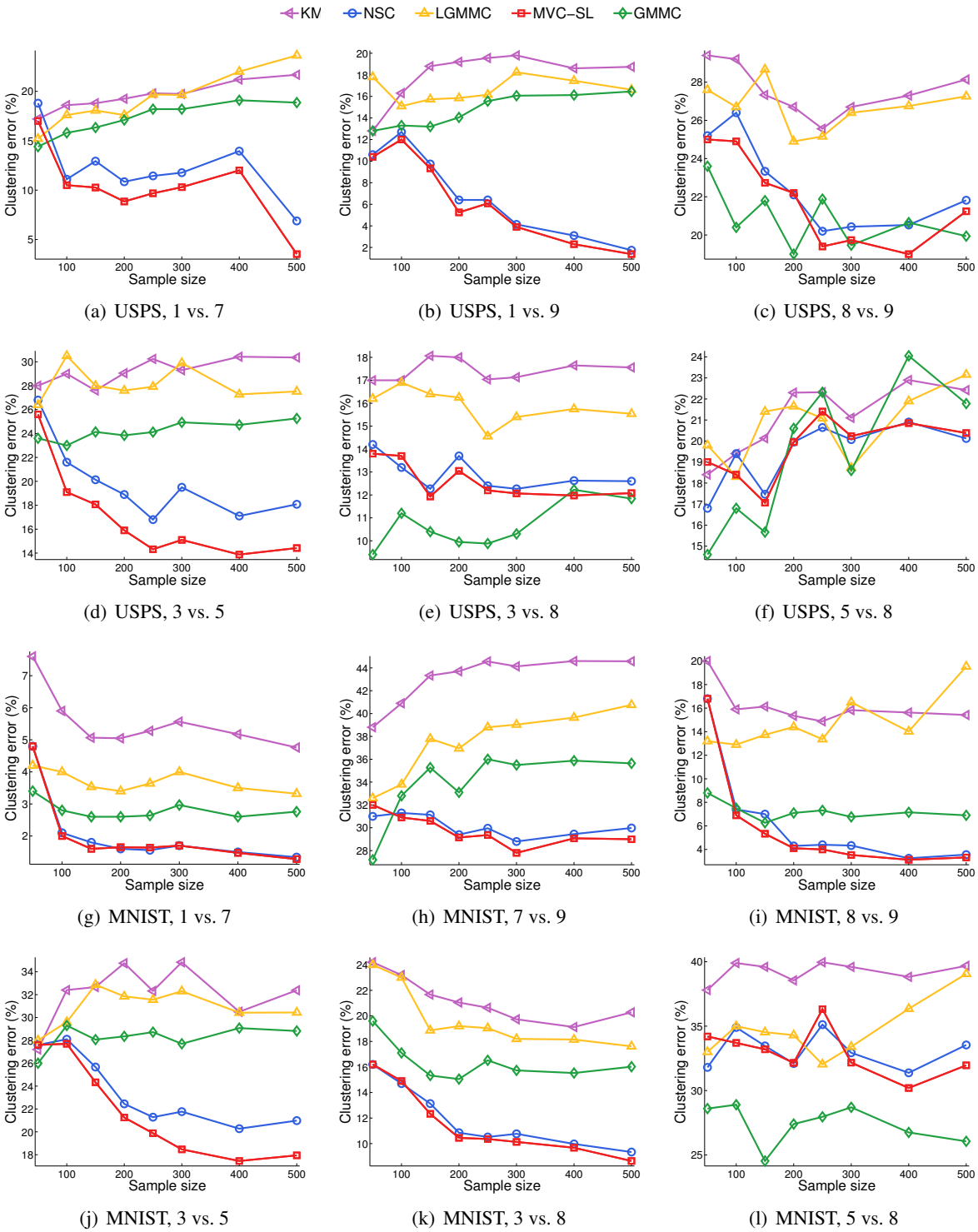
Secondly, we take the images of handwritten digits in USPS and MNIST. Instead of testing KM, NSC, LGMMC, GMMC and MVC-SL on all forty-five pairwise clustering tasks, a few challenging tasks were selected, namely, the pairs $\{1,7\},\{1,9\},\{8,9\},\{3,5\},\{3,8\},\{5,8\}$ of USPS and $\{1,7\},\{7,9\},\{8,9\},\{3,5\},\{3,8\},\{5,8\}$ of MNIST. The task digits 7 vs. 9 of USPS is too hard for all algorithms and then we selected an easier task digits 1 vs. 9. Unlike the training data in the IDA benchmark repository that are already standardized (i.e., normalized to mean zero and standard deviation one) by the provider, the 8-bit gray-scale images in USPS/MNIST are raw data represented by 256-/784-dimensional vectors of integers between 0 and 255. The popular pre-processing is to divide each integer by 255 and thus change the representation to vectors of floating-point numbers between 0 and 1. As a consequence, $\langle x_i, x_j \rangle$ is always nonnegative for any $1 \leq i, j \leq n$ and we can use the cosine similarity for NSC, where in our experiments the hyperparameter k of the k-nearest neighbors was the best value among $\{3,4,5,6,7,8\}$ for each random sampling. The same cosine similarity was also applied to MVC-SL. However, this cosine similarity did not work for the other three algorithms here, and then we still used the Gaussian similarity with σ as the best value among $\{4m_\sigma, 2m_\sigma, m_\sigma, m_\sigma/2, m_\sigma/4\}$ for each random sampling, where m_σ was the mean pairwise distance defined in Equation (27). The settings for other hyperparameters of LGMMC, GMMC, and MVC-SL were exactly same as the experiments on the artificial data sets.

Figure 7 reports the means of the clustering error by each algorithm on each task. The sample sizes were $\{50,100,150,200,250,300,400,500\}$ for all tasks, and each mean value was obtained by repeatedly running an algorithm on 10 random samplings. Given a certain task with sample size n, we first merged all data of the two classes and then randomly sampled a subset of size n, so the classes in the resulting subset were not necessarily balanced when n was small. Moreover, Table 3 summarizes the means with standard errors of the clustering error, in which each algorithm has 80 random samplings on each task. Since the sample sizes here varied in a large range, we performed the paired t-test of the null hypothesis that the difference of the clustering error is from a Gaussian distribution with mean zero and unknown variance, against the alternative hypothesis that the mean is not zero.

We can see from Figure 7 that the easiest task is MNIST 1 vs. 7, such that the mean clustering errors of MVC-SL and NSC were less than two percents when $n \geq 100$, and the hardest tasks are MNIST 7 vs. 9 and 5 vs. 8, where no algorithm was better than twenty-five percents. Both Figure 7 and Table 3 show that the relatively easy tasks include the pairs $\{1,7\},\{1,9\},\{3,8\}$ of USPS and $\{1,7\},\{8,9\},\{3,8\}$ of MNIST, while the relatively hard tasks are the pairs $\{8,9\},\{3,5\},\{5,8\}$ of USPS and $\{7,9\},\{3,5\},\{5,8\}$ of MNIST. In addition, according to Figure 7, the mean clustering errors of MVC-SL were basically non-increasing except in panel (f) USPS 5 vs. 8, and MVC-SL, NSC and GMMC usually outperformed KM and LGMMC, as in Table 3. Similarly, MVC-SL was either the best algorithm or comparable to the best algorithm on ten out of twelve tasks according to Table 3, among which it was best on eight tasks and outperformed all others on seven tasks. The second best algorithm GMMC was best on four tasks, and then NSC was comparable on two tasks. In a word, MVC-SL was fairly promising on USPS and MNIST.

8.3.3 NEWSGROUP DOCUMENTS

The benchmark 20Newsgroups has three versions containing 19997, 18846, and 18828 newsgroup documents, partitioned nearly evenly across twenty different newsgroups. The second version with



(a) USPS, 1 vs. 7 (b) USPS, 1 vs. 9 (c) USPS, 8 vs. 9

(d) USPS, 3 vs. 5 (e) USPS, 3 vs. 8 (f) USPS, 5 vs. 8

(g) MNIST, 1 vs. 7 (h) MNIST, 7 vs. 9 (i) MNIST, 8 vs. 9

(j) MNIST, 3 vs. 5 (k) MNIST, 3 vs. 8 (l) MNIST, 5 vs. 8

Figure 7: Means of the clustering error (in %) on USPS and MNIST.

	KM	NSC	LGMMC	MVC-SL	GMMC
USPS, 1 vs. 7	19.5 ± 0.36	12.2 ± 0.91	19.2 ± 0.68	$\mathbf{10.3 \pm 0.89}$	17.3 ± 0.37
USPS, 1 vs. 9	18.0 ± 0.55	6.9 ± 0.85	16.6 ± 0.54	$\mathbf{6.3 \pm 0.77}$	14.7 ± 0.38
USPS, 8 vs. 9	27.5 ± 0.72	22.5 ± 0.89	26.7 ± 0.87	$\mathbf{21.8 \pm 0.93}$	$\mathbf{20.8 \pm 0.80}$
USPS, 3 vs. 5	29.2 ± 0.61	19.9 ± 0.97	28.1 ± 0.72	$\mathbf{17.0 \pm 0.96}$	24.2 ± 0.62
USPS, 3 vs. 8	17.4 ± 0.52	12.9 ± 0.46	15.9 ± 0.47	12.6 ± 0.47	$\mathbf{10.6 \pm 0.48}$
USPS, 5 vs. 8	21.1 ± 0.65	$\mathbf{19.4 \pm 0.59}$	20.8 ± 0.74	$\mathbf{19.7 \pm 0.67}$	$\mathbf{19.3 \pm 0.84}$
MNIST, 1 vs. 7	5.5 ± 0.36	$\mathbf{2.1 \pm 0.35}$	3.7 ± 0.21	$\mathbf{2.0 \pm 0.35}$	2.8 ± 0.19
MNIST, 7 vs. 9	43.1 ± 0.44	30.1 ± 0.59	37.4 ± 0.58	$\mathbf{29.7 \pm 0.62}$	33.9 ± 0.56
MNIST, 8 vs. 9	16.1 ± 0.96	6.4 ± 0.77	14.7 ± 0.80	$\mathbf{5.9 \pm 0.75}$	7.2 ± 0.36
MNIST, 3 vs. 5	32.1 ± 0.65	23.5 ± 0.66	30.9 ± 0.52	$\mathbf{21.8 \pm 0.72}$	28.3 ± 0.47
MNIST, 3 vs. 8	21.2 ± 0.49	11.9 ± 0.54	19.8 ± 0.58	$\mathbf{11.6 \pm 0.59}$	16.4 ± 0.54
MNIST, 5 vs. 8	39.2 ± 0.47	33.2 ± 1.17	34.7 ± 0.79	33.0 ± 1.22	$\mathbf{27.4 \pm 0.80}$

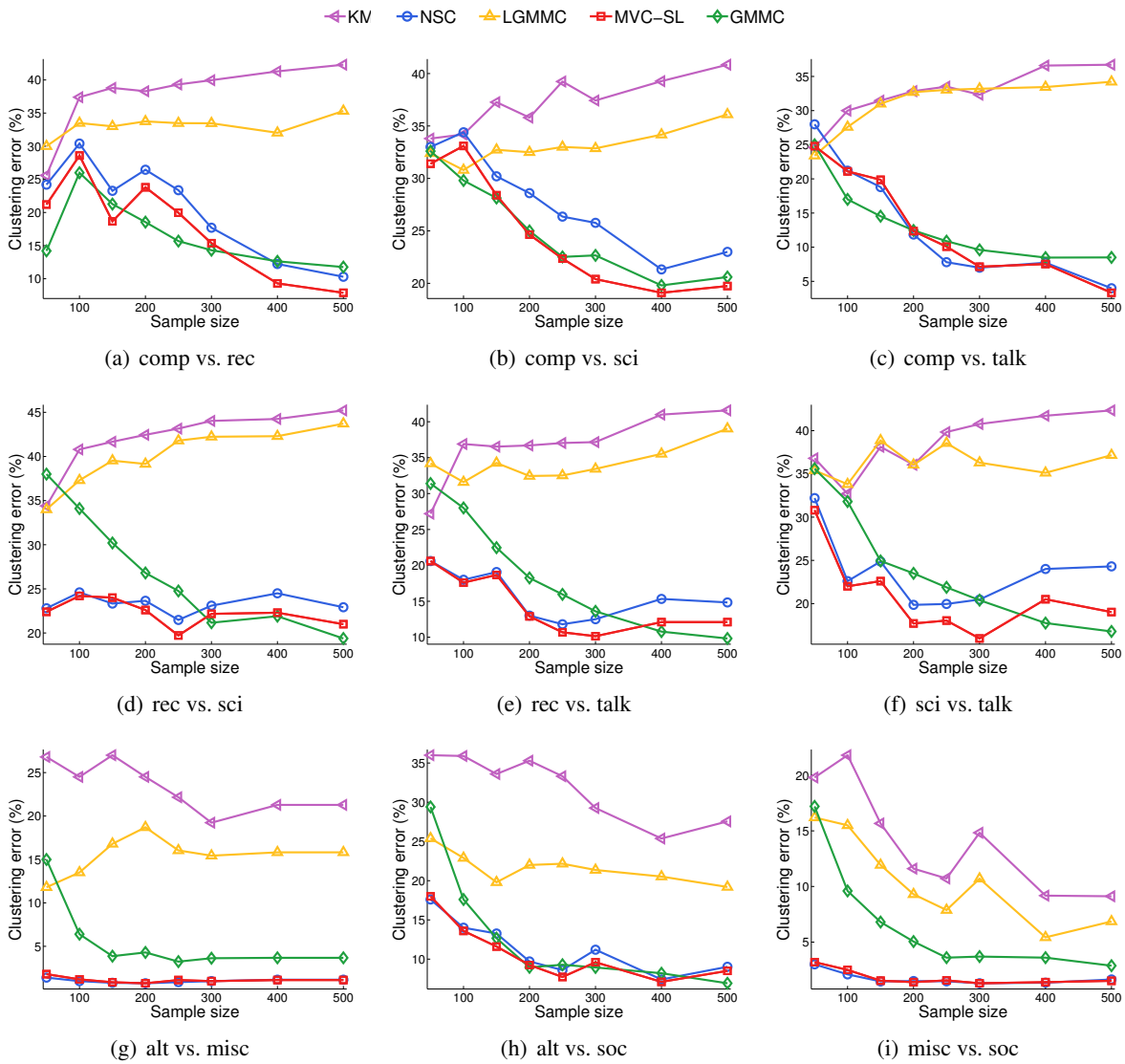
Table 3: Means with standard errors of the clustering error (in %) on USPS and MNIST. For each task, the best algorithm and comparable ones based on the paired t-test at the significance level 5% are highlighted in boldface.

18846 documents is recommended by the original provider[15] and hence is used in our experiments. The documents in 20Newsgroups can be further grouped into seven topics: They are 'alt', 'comp', 'misc', 'rec', 'sci', 'soc' and 'talk', with 799, 4891, 975, 3979, 3952, 997 and 3253 documents respectively, where comp consists of five classes, each of rec, sci and talk consists of four classes, and each of alt, misc and soc consists of a single class. We prepared nine pairwise clustering tasks which included all tasks between the four multi-modal topics and all tasks between the three uni-modal topics. The term-frequency vectors were processed into term-frequency-inverse-document-frequency vectors using the script written by the provider[16] for the whole data set. We tried all of the three similarity measures, and found that for any algorithm no one was consistently better than the other two. However, the locally-scaled similarity generally fitted all five algorithms, where the hyperparameter k was the best value in $\{3,4,5,6,7,8\}$ for each random sampling. The settings for other hyperparameters of LGMMC, GMMC and MVC-SL were exactly same as the experiments on the artificial data sets.

Figure 8 reports the means of the clustering error by each algorithm on each task. The sample sizes were $\{50,100,150,200,250,300,400,500\}$ for all tasks, and each mean value was averaged over 10 random samplings. Similarly to the random samplings of USPS and MNIST, the classes in each random sampling here were not necessarily balanced when n was small. In addition, Table 4 summarizes the means with standard errors of the clustering error, in which each algorithm has 80 random samplings on each task. The paired t-test was performed due to the varied sample sizes.

We can see from Figure 8 and Table 4 that the tasks between the four multi-modal topics are more difficult than the tasks between the three uni-modal topics. Two tasks involving misc (i.e., alt vs. misc and misc vs. soc) are easiest, and three tasks involving sci (i.e., comp vs. sci, rec vs. sci, and sci vs. talk) are hardest. Moreover, MVC-SL, NSC and GMMC usually outperformed KM and LGMMC, and Figure 8 also illustrates that the mean clustering errors of MVC-SL were basically non-increasing. As shown in Table 4, MVC-SL was either the best algorithm or comparable to the

15. See `http://qwone.com/~jason/20Newsgroups/`.
16. See `http://www.cad.zju.edu.cn/home/dengcai/Data/code/tfidf.m`.



(a) comp vs. rec (b) comp vs. sci (c) comp vs. talk

(d) rec vs. sci (e) rec vs. talk (f) sci vs. talk

(g) alt vs. misc (h) alt vs. soc (i) misc vs. soc

Figure 8: Means of the clustering error (in %) on 20Newsgroups.

best algorithm on eight out of nine tasks, among which it was best on six tasks and outperformed all others on four tasks. The second best algorithm NSC was best on three tasks, and then GMMC was best on two tasks and comparable on one task. In a word, MVC-SL was also fairly promising on 20Newsgroups.

8.3.4 ISOLATED SPOKEN LETTERS

The final benchmark is Isolet from the *UCI machine learning repository*. The data were collected by letting 150 subjects speak the name of each letter of the alphabet twice, while two 'F' and one 'M' were dropped due to difficulties in recording. Unlike the features of the previous benchmarks USPS, MNIST and 20Newsgroups, the acoustic features of Isolet are extracted by different ways and possess different physical meanings, including spectral coefficients, contour features, sonorant

	KM	NSC	LGMMC	MVC-SL	GMMC
comp vs. rec	37.9 ± 0.77	21.0 ± 1.46	33.1 ± 0.57	$\mathbf{18.1 \pm 1.41}$	$\mathbf{16.8 \pm 0.74}$
comp vs. sci	37.2 ± 0.65	27.8 ± 1.20	33.1 ± 0.61	$\mathbf{24.9 \pm 1.17}$	$\mathbf{25.1 \pm 0.69}$
comp vs. talk	32.3 ± 0.93	$\mathbf{13.3 \pm 1.69}$	31.1 ± 0.73	$\mathbf{13.3 \pm 1.65}$	$\mathbf{13.3 \pm 0.80}$
rec vs. sci	42.0 ± 0.55	23.3 ± 0.84	40.0 ± 0.73	$\mathbf{22.3 \pm 0.85}$	27.0 ± 1.01
rec vs. talk	36.8 ± 0.76	15.6 ± 1.11	34.1 ± 1.02	$\mathbf{14.3 \pm 1.08}$	18.8 ± 1.08
sci vs. talk	38.5 ± 0.71	23.5 ± 1.01	36.4 ± 0.67	$\mathbf{20.8 \pm 0.97}$	24.1 ± 0.86
alt vs. misc	23.3 ± 1.85	$\mathbf{1.0 \pm 0.12}$	15.5 ± 1.07	1.1 ± 0.13	5.5 ± 0.60
alt vs. soc	32.0 ± 1.05	11.3 ± 1.01	21.7 ± 0.95	$\mathbf{10.7 \pm 0.85}$	12.7 ± 0.91
misc vs. soc	14.1 ± 1.32	$\mathbf{1.7 \pm 0.16}$	10.5 ± 0.67	$\mathbf{1.8 \pm 0.16}$	6.6 ± 0.60

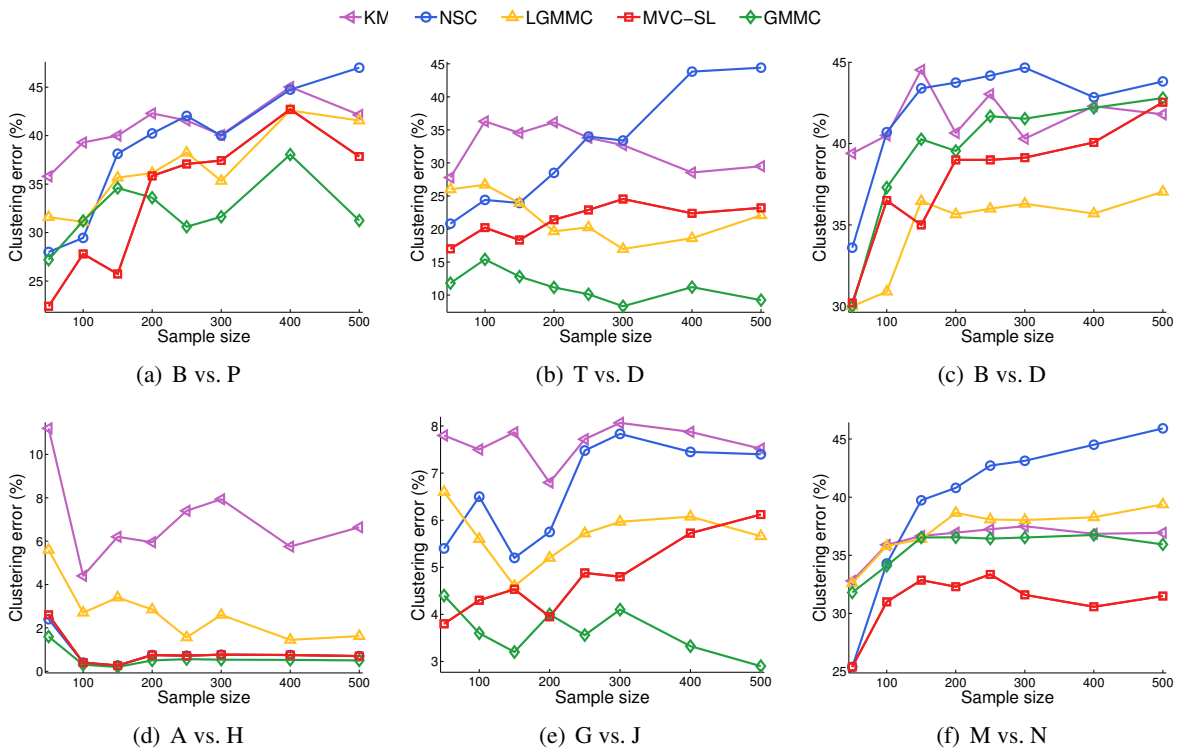
Table 4: Means with standard errors of the clustering error (in %) on 20Newsgroups. For each task, the best algorithm and comparable ones based on the paired t-test at the significance level 5% are highlighted in boldface.

	KM	NSC	LGMMC	MVC-SL	GMMC
B vs. P	40.8 ± 0.64	38.7 ± 1.04	36.5 ± 0.86	$\mathbf{33.4 \pm 1.25}$	$\mathbf{32.3 \pm 1.30}$
T vs. D	32.4 ± 0.93	31.7 ± 1.48	21.8 ± 1.24	21.2 ± 1.02	$\mathbf{11.2 \pm 1.09}$
B vs. D	41.6 ± 0.55	42.1 ± 0.63	$\mathbf{34.8 \pm 0.73}$	37.7 ± 0.82	39.4 ± 0.73
A vs. H	6.9 ± 0.68	0.8 ± 0.19	2.7 ± 0.41	0.9 ± 0.21	$\mathbf{0.6 \pm 0.15}$
G vs. J	7.6 ± 0.32	6.6 ± 0.72	5.7 ± 0.28	4.8 ± 0.28	$\mathbf{3.6 \pm 0.22}$
M vs. N	36.4 ± 0.49	39.6 ± 0.87	37.2 ± 0.47	$\mathbf{31.1 \pm 0.64}$	35.6 ± 0.47

Table 5: Means with standard errors of the clustering error (in %) on Isolet. For each task, the best algorithm and comparable ones based on the paired t-test at the significance level 5% are highlighted in boldface.

features, pre-sonorant features and post-sonorant features. All features are real-valued and scaled into the range -1 to $+1$. Generally speaking, all five algorithms can easily deal with the majority of pairwise clustering tasks, if we randomly choose two letters. Therefore, similarly to USPS and MNIST, a few challenging tasks that might sometimes be difficult for the mankind were selected: The letters B vs. P, T vs. D, B vs. D, A vs. H, G vs. J, and M vs. N. The hyperparameters here were slightly different from the previous experiments for better performance. The cosine similarity was applied to NSC, and the hyperparameter k was the best value in $\{1,2,3,4,5,6\}$ for each random sampling. The Gaussian similarity was still used for KM, LGMMC and GMMC, and the hyperparameter σ was the best value in $\{2m_\sigma, m_\sigma, m_\sigma/2, m_\sigma/4, m_\sigma/8\}$ for each random sampling, where m_σ was defined in Equation (27). For MVC-SL, we adopted either $Q = L_{\mathrm{sym}} + I_n/n$ where L_{sym} was constructed from the cosine similarity or $Q = nI_n - W$ with the Gaussian similarity depending on the task and the sample size n, and the hyperparameter k or σ was chosen in the same way. A key observation here was that for certain tasks such as M vs. N, the former specification was preferable for small n, whereas the latter specification was more advisable for relatively large n. The settings for other hyperparameters were exactly same as the experiments on the artificial data sets.

Figure 9 reports the means of the clustering error by each algorithm on each task. The sample sizes were $\{50,100,150,200,250,300,400,500\}$ for all tasks, and each mean value was averaged



(a) B vs. P (b) T vs. D (c) B vs. D

(d) A vs. H (e) G vs. J (f) M vs. N

Figure 9: Means of the clustering error (in %) on Isolet.

over 10 random samplings. Similarly to the random samplings of USPS and MNIST, the classes in each random sampling here were not necessarily balanced when n was small. In addition, Table 5 summarizes the means with standard errors of the clustering error, in which each algorithm has 80 random samplings on each task. The paired t-test was performed due to the varied sample sizes.

We can see from Figure 9 and Table 5 that the tasks A vs. H and G vs. J are very easy, and the tasks B vs. P, B vs. D and M vs. N are very hard. Interestingly, T vs. D is much easier than B vs. P and B vs. D, such that the lowest mean clustering errors on B vs. P and B vs. D were almost three times larger than the lowest mean clustering error on T vs. D. Unlike the curves shown in Figures 7 and 8, the mean clustering errors of MVC-SL in Figure 9 were basically non-increasing only in panel (d) A vs. H. Furthermore, LGMMC instead of NSC became a competitive algorithm besides GMMC and MVC-SL in Table 5, unlike the performance in Tables 3 and 4. According to Table 5, GMMC was the best algorithm on four tasks, MVC-SL was best on one task and also comparable to the best algorithm on one task, and LGMMC was best on one task. Nevertheless, MVC-SL was still satisfying on Isolet, if considering that MVC-SL consumed less than five percents of the total computation time while GMMC consumed over ninety percents, and thus GMMC was remarkably less computationally-efficient than MVC-SL.

9. Conclusions

We proposed a new discriminative clustering model called maximum volume clustering (MVC) to partition the data samples into two clusters based on the large volume principle. Two algorithms to

approximate the basic model of MVC were developed: MVC-HL relaxes MVC to a semi-definite programming problem that is convex but time-consuming; MVC-SL employs sequential quadratic programming that is non-convex but computationally-efficient. Then, we demonstrated that MVC includes the optimization problems of some well-known clustering methods as special limit cases, and discussed the finite sample stability and the clustering error bound of MVC-SL in great detail. Based on the encouraging experimental results on three artificial and fourteen benchmark data sets, we conclude that the proposed MVC approach is promising, especially for images and text.

The future work includes but is not limited to the following three directions: Multi-way extension, improved optimization, and model selection and specification of Q. We briefly discuss these future directions below.

First of all, the basic model of MVC is currently binary, and it needs a multi-way extension to partition the data samples into more than two clusters. To this end, we should extend the definition of the volume before extending the basic model of MVC. Unlike the margin, there exists no multi-class definition of the volume hitherto. We may borrow the idea of the multi-class definition of the margin in Crammer and Singer (2001) based on which the first multi-way extension of MMC was proposed (Xu and Schuurmans, 2005).

Secondly, the proposed approximation schemes and optimization algorithms for MVC may be improved. However, we believe that the improvement cannot be straightforward. We have considered several options and found that none of them befits MVC well. Recall that the primal problem of MVC-SL defined in (4) is non-convex, and the *concave-convex procedure* and *constrained concave-convex procedure* (CCCP) (Yuille and Rangarajan, 2003; Smola et al., 2005) seem able to solve it. In fact, the former technique can only be applied to the Lagrange function $L(\boldsymbol{h},\eta)$, and η as an optimization variable may diverge even though $\boldsymbol{h}$ is guaranteed to converge given constant η. On the other hand, the latter technique accepts any first-order equality constraint and any inequality constraint involving the difference of two convex functions, but the second-order equality constraint like $\boldsymbol{h}^\top\boldsymbol{h}=1$ is unacceptable. If we relax the equality constraint $\boldsymbol{h}^\top\boldsymbol{h}=1$ into an inequality constraint $\boldsymbol{h}^\top\boldsymbol{h}\leq 1$, we will get

$$\min_{\boldsymbol{h}\in\mathbb{R}^n} \; -2\|\boldsymbol{h}\|_1+\gamma\boldsymbol{h}^\top Q\boldsymbol{h} \qquad \text{s.t. } \boldsymbol{h}^\top\boldsymbol{h}\leq 1. \tag{28}$$

Unfortunately, CCCP fails to solve optimization (28) again, since now we cannot assume that $\|\boldsymbol{h}\|_1$ is differentiable, and then we cannot easily linearize the concave part of the energy function. Note that the popular trick to cope with ℓ_1-regularization is futile here, since (28) is never equivalent to

$$\begin{aligned}\min_{\boldsymbol{h}\in\mathbb{R}^n} \; & -2\boldsymbol{\alpha}^\top\mathbf{1}_n+\gamma\boldsymbol{h}^\top Q\boldsymbol{h}\\ \text{s.t. } & \boldsymbol{h}^\top\boldsymbol{h}\leq 1,-\boldsymbol{\alpha}\leq\boldsymbol{h}\leq\boldsymbol{\alpha},\boldsymbol{\alpha}\geq\mathbf{0}_n.\end{aligned}$$

Similarly, (28) itself is not *quadratically-constrained quadratic programming* (QCQP) (Boyd and Vandenberghe, 2004) due to the minimization of negative ℓ_1-norm, but it can be reformulated as a QCQP with an optimization variable essentially in $\mathbb{R}^{2n}$:

$$\min_{\boldsymbol{y}\in[-1,+1]^n}\min_{\boldsymbol{h}\in\mathbb{R}^n} \; -2\boldsymbol{h}^\top\boldsymbol{y}+\gamma\boldsymbol{h}^\top Q\boldsymbol{h} \qquad \text{s.t. } \boldsymbol{h}^\top\boldsymbol{h}\leq 1. \tag{29}$$

Although optimization (29) is convex in $\boldsymbol{y}$ and convex in $\boldsymbol{h}$, it is not jointly convex in $\boldsymbol{y}$ and $\boldsymbol{h}$, so no off-the-shelf QCQP solver is applicable and we need relax it via semi-definite programming or *reformulation-linearization technique* (Sherali and Adams, 1998) once more. Actually, the feasible

region $[-1,+1]^n$ of $\boldsymbol{y}$ is as difficult as the combinatorial $\{-1,+1\}^n$, and all of optimizations (2), (4), (28) and (29) are NP-hard, regardless of the different feasible regions of $\boldsymbol{h}$. That being said, the current implementation using sequential quadratic programming is imperfect as the final $\boldsymbol{h}^*$ is a bit sensitive to the initial $\boldsymbol{h}_0$ (see the experimental results reported in Figure 6 for details).

In contrast to MVC-SL, there is much more room for MVC-HL to be improved. GMMC uses a tricky substitution to get (23), and that substitution is so specific that it does not work for MVC-HL. Following the idea of LGMMC, we can obtain an alternative relaxation as

$$\begin{aligned}\min_{\boldsymbol{\mu}\in\mathbb{R}^{2^n}} \min_{\boldsymbol{\alpha}} \; & -2\boldsymbol{\alpha}^\top \mathbf{1}_n + \gamma\boldsymbol{\alpha}^\top \left(\sum\nolimits_{t:-b\le \boldsymbol{y}_t^\top \mathbf{1}_n \le b} \mu_t Q \circ \boldsymbol{y}_t \boldsymbol{y}_t^\top\right) \boldsymbol{\alpha} \\ \text{s.t.} \; & \boldsymbol{\mu}^\top \mathbf{1}_{2^n} = 1, \boldsymbol{\mu} \ge \mathbf{0}_{2^n} \\ & \boldsymbol{\alpha}^\top \boldsymbol{\alpha} = 1, \boldsymbol{\alpha} \ge \mathbf{0}_n.\end{aligned}$$

Similarly, this optimization can also be regarded as a multiple kernel learning problem and solved by the cutting plane method, as LGMMC. However, the inner optimization subproblem is difficult due to $\boldsymbol{\alpha}^\top\boldsymbol{\alpha} = 1$ instead of $\boldsymbol{\alpha}^\top\mathbf{1}_n = 1$ in LGMMC, and we decide to investigate how to solve it in our future study since MVC-HL is not the main focus of the current paper.

Thirdly, in our experiments we always use the best candidate hyperparameters in the hindsight, since there lacks a systematic way to tune the hyperparameters for clustering. Such choices may be acceptable from the theoretical standpoint but not enough from the practical standpoint. Notice that any (cross-) validation technique using the clustering error, which is the in-sample test error on the same data to be clustered, simply does not work for model selection. In order to do model selection, a criterion other than the clustering error is necessary. Fortunately, a few information criteria exist though they are not uniformly effective for all clustering algorithms. In Sugiyama et al. (2011), the *mutual information* (MI) (Shannon, 1948) was used for *MI based clustering* (Gomes et al., 2010) via *maximum likelihood MI* (Suzuki et al., 2008) for model selection, and *squared-loss MI* (Suzuki et al., 2009) was used for *squared-loss MI based clustering* (Sugiyama et al., 2011) via *least-squares MI* (Suzuki et al., 2009) for model selection.

What is more, it is unclear how to specify the input matrix Q appropriately for a given data set, including a proper similarity measure and the construction of Q from it. According to von Luxburg et al. (2012), the former issue is actually open for all existing clustering algorithms and it probably has no uniformly effective solution. For the latter issue, we suggest MVC-SL with $Q = L_{\text{sym}} + I_n/n$, where L_{sym} is the normalized graph Laplacian, and the underlying similarity measure can be any similarity suitable for spectral clustering. Then, it is still unsolved when we should use MVC-SL, and when we should use the family of MMC or other clustering algorithms. Unfortunately, there is no answer from a theoretical point of view since clustering has no supervision at all. Nevertheless, MVC-SL may work with high probability in practice when spectral clustering works. We argue that it may be the minimization of negative ℓ_1-norm in MVC-SL that has improved spectral clustering as shown in panel (c) of Figure 6. Its preference of non-sparse optimal solutions may lead to a better approximation to the normalized cut criterion (Shi and Malik, 2000) than spectral clustering.

Acknowledgments

The authors would like to thank anonymous reviewers for the helpful comments. GN is supported by the MEXT scholarship No. 103250, LS is supported by the NSFC programs No. 61170180 and

No. 61035003, and MS is supported by the FIRST program. The preliminary work has been done when GN was studying at Department of Computer Science and Technology, Nanjing University, and BD was studying at Institute of Automation, Chinese Academy of Sciences.

Appendix A. Proofs of Theoretical Results in Section 5.2

In this appendix, we prove the lemmas and theorems appeared in Section 5.2.

A.1 Proof of Lemma 9

If $\exists j \in \{1,\dots,n\}$, $\boldsymbol{e}_j$ or $-\boldsymbol{e}_j$ is an eigenvector of Q, there should exist an eigenvalue $\lambda > 0$ such that $Q\boldsymbol{e}_j = \lambda \boldsymbol{e}_j$. This equation means that $Q_{j,j} = \lambda$ and $\forall i \neq j, Q_{i,j} = 0$. In other words, x_j is isolated and X_n is reducible. ■

A.2 Proof of Theorem 10

If x_i is isolated in X_n, let $\delta_1 = \cdots = \delta_n = 1$, $\mathcal{K} = \{i\}$ and by definition X_n is SI-symmetric.

If X_n is axisymmetric under a permutation ϕ, without loss of generality, we assume $\phi(1) = 2$ and let $\delta_1 = -1, \delta_2 = \cdots = \delta_n = 1$ and $\mathcal{K} = \{1,2\}$. Then X_n is SI-symmetric by Equation (20),

$$\left(\sum\nolimits_{k\in\mathcal{K}} \delta_k \boldsymbol{e}_k\right)^\top Q \left(\sum\nolimits_{k\notin\mathcal{K}} \delta_k \boldsymbol{e}_k\right) = \sum_{i=3}^{n} (Q_{2,i} - Q_{1,i}) = 0,$$

since $\forall i \in \{3,\dots,n\}$, $\phi(i) \notin \{1,2\}$ and $Q_{1,i} = Q_{2,\phi(i)}$. ■

A.3 Proof of Theorem 11

When $n = 2$, X_2 must be axisymmetric if $Q_{1,1} = Q_{2,2}$, and we can know that X_2 is SI-symmetric by Theorem 10.

When $n > 2$, assume that X_n is irreducible due to Theorem 10, and then $\mathbb{R}^n$ has two disjoint bases according to Lemma 9: The standard basis and the set of the principle axes of $\mathcal{E}(\mathcal{H}_Q)$. We present an indirect proof of the theorem as follows.

Step 1. Let $\lambda_1,\dots,\lambda_n$ be the eigenvalues of Q and $\boldsymbol{v}_1,\dots,\boldsymbol{v}_n$ be the associated normalized eigenvectors. Suppose that $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ are the directions of two principal axes of $\mathcal{E}(\mathcal{H}_Q)$ with the same length $1/\sqrt{\lambda_i} = 1/\sqrt{\lambda_j}$. There should be at least one principal axis $\boldsymbol{v}_l$ such that $l \notin \{i,j\}$, $\lambda_l \neq \lambda_i$, and $\mathcal{E}(\mathcal{H}_Q)$ is rotational about $\boldsymbol{v}_l$ along the circle

$$C(\boldsymbol{v}_i, \boldsymbol{v}_j) := \{\cos(\theta)\boldsymbol{v}_i + \sin(\theta)\boldsymbol{v}_j \mid \theta \in [0, 2\pi)\}.$$

Otherwise, all principal axes have the same length and thus $\mathcal{E}(\mathcal{H}_Q)$ is a perfect ball, which contradicts the fact that $\boldsymbol{e}_1,\dots,\boldsymbol{e}_n$ are not eigenvectors of Q.

Further suppose that $\lambda_k \neq \lambda_l$ for any $k \neq l$, that is, the principal axis with the direction $\boldsymbol{v}_l$ has a unique length. As a consequence, $\boldsymbol{v}_l$ has a fixed position and cannot rotate within $C(\boldsymbol{v}_k, \boldsymbol{v}_l)$ for any $k \notin \{i,j,l\}$. Otherwise, all vectors in $C(\boldsymbol{v}_k, \boldsymbol{v}_l)$ are legal principal axes and can be considered as $\boldsymbol{v}_l$ with a fixed position.

We can know that $\mathcal{E}(\mathcal{H}_Q)$ intersects the k-th coordinate axis at $\pm\boldsymbol{e}_k/\sqrt{\kappa}$ from $Q_{k,k} = \kappa$, and the intersections compose an $(n-1)$-dimensional hyperplane. Principal axes of $\mathcal{E}(\mathcal{H}_Q)$ are orthogonal

and have at most $(n-1)$ distinct lengths, and $\mathcal{E}(\mathcal{H}_Q)$ also has a set of n orthogonal axes with the same length $1/\sqrt{\kappa}$, that is, the set $\{\boldsymbol{e}_1/\sqrt{\kappa},\ldots,\boldsymbol{e}_n/\sqrt{\kappa}\}$. Hence, any principal axis in a fixed position, especially $\boldsymbol{v}_l$, should lie on the central direction of a certain quadrant with dimensionality at least two. In other words, $\boldsymbol{v}_l$ can be written in the form of

$$\boldsymbol{v}_l = \frac{1}{\sqrt{\sum_{k=1}^n \delta_k^2}} \sum_{k=1}^n \delta_k \boldsymbol{e}_k, \quad \delta_k \in \{-1,0,1\},$$

where $\delta_1,\ldots,\delta_n$ cannot be all zeros.

Step 2. Let $\mathcal{K} = \{k \mid \delta_k = 0\}$ and one has $0 \le \#\mathcal{K} < n$ where $\#$ measures the cardinality. We discuss the cases $\#\mathcal{K} > 0$ and $\#\mathcal{K} = 0$ separately.

If $\#\mathcal{K} > 0$, we reset $\delta_k = 1$ for $k \in \mathcal{K}$. Subsequently,

$$\begin{aligned}
\left(\sum\nolimits_{k\in\mathcal{K}} \delta_k \boldsymbol{e}_k\right)^\top Q \left(\sum\nolimits_{k\notin\mathcal{K}} \delta_k \boldsymbol{e}_k\right) &= \left(\sum\nolimits_{k\in\mathcal{K}} \boldsymbol{e}_k\right)^\top Q \left(\sqrt{n-\#\mathcal{K}}\boldsymbol{v}_l\right) \\
&= \left(\sum\nolimits_{k\in\mathcal{K}} \boldsymbol{e}_k\right)^\top \sqrt{n-\#\mathcal{K}}(Q\boldsymbol{v}_l) \\
&= \left(\sum\nolimits_{k\in\mathcal{K}} \boldsymbol{e}_k\right)^\top \sqrt{n-\#\mathcal{K}}(\lambda_l \boldsymbol{v}_l) \\
&= \lambda_l \left(\sum\nolimits_{k\in\mathcal{K}} \boldsymbol{e}_k\right)^\top \left(\sqrt{n-\#\mathcal{K}}\boldsymbol{v}_l\right) \\
&= \lambda_l \left(\sum\nolimits_{k\in\mathcal{K}} \boldsymbol{e}_k\right)^\top \left(\sum\nolimits_{k\notin\mathcal{K}} \delta_k \boldsymbol{e}_k\right) \\
&= \lambda_l \sum_{k\in\mathcal{K},k'\notin\mathcal{K}} \delta_{k'} \boldsymbol{e}_k^\top \boldsymbol{e}_{k'} \\
&= 0,
\end{aligned}$$

due to $Q\boldsymbol{v}_l = \lambda_l \boldsymbol{v}_l$ and the orthonormal condition of the basis $\{\boldsymbol{e}_1,\ldots,\boldsymbol{e}_n\}$. If $\#\mathcal{K} = 0$, without loss of generality, assume that $\delta_1 = -\delta_2 = 1$ since $n > 2$ and the sign of $\boldsymbol{v}_l$ is arbitrary. The first two rows of the eigenvalue equation $Q\boldsymbol{v}_l = \lambda_l \boldsymbol{v}_l$ tell us

$$\begin{cases}
\kappa - Q_{1,2} + \sum_{k=3}^n \delta_k Q_{1,k} = \lambda_l \\
Q_{2,1} - \kappa - \sum_{k=3}^n \delta_k Q_{2,k} = -\lambda_l
\end{cases} \quad \Rightarrow \quad \sum_{k=3}^n \delta_k (Q_{1,k} - Q_{2,k}) = 0.$$

Hence by resetting $\mathcal{K} = \{1,2\}$, we obtain

$$\left(\sum\nolimits_{k\in\mathcal{K}} \delta_k \boldsymbol{e}_k\right)^\top Q \left(\sum\nolimits_{k\notin\mathcal{K}} \delta_k \boldsymbol{e}_k\right) = \sum_{k=3}^n \delta_k (Q_{1,k} - Q_{2,k}) = 0.$$

Both cases lead to a contradiction since X_n is SI-asymmetric.

Therefore, all principal axes of $\mathcal{E}(\mathcal{H}_Q)$ have distinct lengths, which is exactly what we were to prove. ■

A.4 Proof of Theorem 12

Let us denote $\boldsymbol{h}^* = (h_1, \ldots, h_n)^\top$ and consider $\boldsymbol{h}^\star = (h_{\phi(1)}, \ldots, h_{\phi(n)})^\top$.

Obviously, $\|\boldsymbol{h}^\star\|_1 = \|\boldsymbol{h}^*\|_1$ and $\|\boldsymbol{h}^\star\|_2 = \|\boldsymbol{h}^*\|_2$. Moreover,

$$\sum_{i,j=1}^{n} Q_{i,j} h_{\phi(i)} h_{\phi(j)} = \sum_{i,j=1}^{n} Q_{\phi(i),\phi(j)} h_{\phi(i)} h_{\phi(j)} = \sum_{k,l=1}^{n} Q_{k,l} h_k h_l,$$

because of the third property of Definition 7. Hence, $\boldsymbol{h}^{\star\top} Q \boldsymbol{h}^\star = \boldsymbol{h}^{*\top} Q \boldsymbol{h}^*$ and then $G(\boldsymbol{h}^\star) = G(\boldsymbol{h}^*)$.

Similarly, $\forall i \in \{1, \ldots, n\}$,

$$\sum_{j=1}^{n} Q_{i,j} h_{\phi(j)} = \sum_{j=1}^{n} Q_{\phi(i),\phi(j)} h_{\phi(j)} = \sum_{k=1}^{n} Q_{\phi(i),k} h_k,$$

where we use the third property of Definition 7 again. As a result,

$$\begin{aligned}
[g(\boldsymbol{h}^\star)]_i &= \gamma \sum_{j=1}^{n} Q_{i,j} h_{\phi(j)} - \eta h_{\phi(i)} - \operatorname{sign}(h_{\phi(i)}) \\
&= \gamma \sum_{k=1}^{n} Q_{\phi(i),k} h_k - \eta h_{\phi(i)} - \operatorname{sign}(h_{\phi(i)}) \\
&= [g(\boldsymbol{h}^*)]_{\phi(i)}.
\end{aligned}$$

Hence, $g(\boldsymbol{h}^\star) = \mathbf{0}_n$ according to the Karush-Kuhn-Tucker condition $g(\boldsymbol{h}^*) = \mathbf{0}_n$, which means that $\boldsymbol{h}^\star$ is also a minimum of optimization (4), since the Hessian matrix $\nabla^2 G(\boldsymbol{h}) = 2(\gamma Q - \eta I_n)$ must be symmetric and positive-definite.

Notice that $d_{\mathcal{H}}(\boldsymbol{h}^*, \boldsymbol{h}^\star) \geq 1$ since $\exists i, h_{\phi(i)} h_i < 0$, with the only exception $d_{\mathcal{H}}(\boldsymbol{h}^*, \boldsymbol{h}^\star) = 0$ when $\operatorname{sign}(\boldsymbol{h}^\star) = -\operatorname{sign}(\boldsymbol{h}^*)$, that is, $\forall i, h_{\phi(i)} h_i < 0$. This completes the proof. ■

A.5 Proof of Theorem 13

We prove the theorem in three steps.

Step 1. Let $0 < \lambda_1 < \cdots < \lambda_n$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the eigenvalues and eigenvectors of Q. Given a minimum $\boldsymbol{h}$, the Karush-Kuhn-Tucker condition $g(\boldsymbol{h}) = \mathbf{0}$ implies that

$$\boldsymbol{h} = \hat{Q}\boldsymbol{y}, \tag{30}$$

where $\boldsymbol{y} = \operatorname{sign}(\boldsymbol{h})$, $\hat{Q} = (\gamma Q - \eta I_n)^{-1}$, and the unknown η satisfies $\eta < \gamma\lambda_1$. Plug Equation (30) into the constraint $\|\boldsymbol{h}\|_2 = 1$, note that $\hat{Q}$ is a symmetric matrix, and then we will have

$$\boldsymbol{y}^\top \hat{Q}^2 \boldsymbol{y} = (\hat{Q}\boldsymbol{y})^\top (\hat{Q}\boldsymbol{y}) = \boldsymbol{h}^\top \boldsymbol{h} = 1. \tag{31}$$

All eigenvalues of Q are different and positive since X_n is anisotropic, so are all eigenvalues of $\hat{Q}$. Consequently, $\hat{Q}^2$ has a unique spectral decomposition. It is easy to see that

$$\boldsymbol{y}^\top \hat{Q}^2 \boldsymbol{y} = \boldsymbol{y}^\top \left(\sum_{i=1}^{n} \mu_i \boldsymbol{v}_i \boldsymbol{v}_i^\top \right) \boldsymbol{y} = \sum_{i=1}^{n} \mu_i (\boldsymbol{v}_i^\top \boldsymbol{y})^2, \tag{32}$$

where $\mu_i = 1/(\gamma\lambda_i - \eta)^2$ is the i-th largest eigenvalue of $\hat{Q}^2$.

Step 2. Define a linear mapping

$$\begin{aligned}\psi : \mathbb{R}^n_\beta &\mapsto \mathbb{R}^n \\ \boldsymbol{\beta} &\mapsto \beta_1 \boldsymbol{v}_1 + \cdots + \beta_n \boldsymbol{v}_n,\end{aligned}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)^\top$, $\mathbb{R}^n_\beta = \mathbb{R}^n$ and we just use the symbol $\mathbb{R}^n_\beta$ to distinguish the domain and the range of ψ. It is obvious that ψ is a vector space automorphism, and the set of vectors $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$ and the set of images $\{\psi(\boldsymbol{e}_1), \ldots, \psi(\boldsymbol{e}_n)\} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ are completely different bases due to Theorem 10 and Lemma 9.

Let $\boldsymbol{\beta} = \psi^{-1}(\boldsymbol{y})$. Then,

$$\|\boldsymbol{y}\|_2 = \sqrt{n} \quad \Rightarrow \quad \beta_1^2 + \cdots + \beta_n^2 = n \tag{33}$$

$$(31) + (32) \quad \Rightarrow \quad \mu_1 \beta_1^2 + \cdots + \mu_n \beta_n^2 = 1. \tag{34}$$

Equation (33) represents a hyper-ball in $\mathbb{R}^n_\beta$, and Equation (34) represents an irrotational ellipsoid in $\mathbb{R}^n_\beta$ since $\mu_1, \ldots, \mu_n$ are distinct eigenvalues. As a result, given any other $\boldsymbol{\beta}' = (\beta'_1, \ldots, \beta'_n)^\top$ satisfying (33) and (34), there exist three disjoint index sets $\mathcal{J}_+, \mathcal{J}_-, \mathcal{J}_0$ such that $\mathcal{J}_+ \cup \mathcal{J}_- \cup \mathcal{J}_0 = \{1, \ldots, n\}$ and

$$\begin{aligned}&\forall j \in \mathcal{J}_+, \beta_j \neq 0, \beta_j + \beta'_j = 0 \\ &\forall j \in \mathcal{J}_-, \beta_j \neq 0, \beta_j - \beta'_j = 0 \\ &\forall j \in \mathcal{J}_0, \beta_j = \beta'_j = 0.\end{aligned}$$

Step 3. For another arbitrarily chosen minimum $\boldsymbol{h}'$ of (4), let $\boldsymbol{y}' = \operatorname{sign}(\boldsymbol{h}')$ and $\boldsymbol{\beta}' = \psi^{-1}(\boldsymbol{y}')$, then $\boldsymbol{\beta}'$ is also a solution to the system of Equations (33) and (34), and it is guaranteed the existence of aforementioned $\mathcal{J}_+, \mathcal{J}_-, \mathcal{J}_0$.

Notice that $\forall j \in \mathcal{J}_+$,

$$\boldsymbol{v}_j^\top (\boldsymbol{y} + \boldsymbol{y}') = \beta_j + \beta'_j = 0 \quad \Rightarrow \quad \boldsymbol{v}_j^\top \boldsymbol{y}' = -\boldsymbol{v}_j^\top \boldsymbol{y}.$$

Similarly, $\forall j \in \mathcal{J}_-, \boldsymbol{v}_j^\top \boldsymbol{y}' = \boldsymbol{v}_j^\top \boldsymbol{y}$ and $\forall j \in \mathcal{J}_0, \boldsymbol{v}_j^\top \boldsymbol{y}' = \boldsymbol{v}_j^\top \boldsymbol{y} = 0$. In a word, we have $(\boldsymbol{v}_j^\top \boldsymbol{y}')^2 = (\boldsymbol{v}_j^\top \boldsymbol{y})^2$ for all $j = 1, \ldots, n$. Hence,

$$\boldsymbol{y}^\top Q \boldsymbol{y} = \sum_{j=1}^n \lambda_j (\boldsymbol{v}_j^\top \boldsymbol{y})^2 = \sum_{j=1}^n \lambda_j (\boldsymbol{v}_j^\top \boldsymbol{y}')^2 = \boldsymbol{y}'^\top Q \boldsymbol{y}',$$

which indicates that $(\boldsymbol{y} + \boldsymbol{y}')^\top Q (\boldsymbol{y} - \boldsymbol{y}') = 0$.

Let $\delta_1 = [\boldsymbol{y}]_1, \cdots, \delta_n = [\boldsymbol{y}]_n$ and $\mathcal{K} = \{k \mid [\boldsymbol{y}]_k = [\boldsymbol{y}']_k, 1 \leq k \leq n\}$. Subsequently, by checking the condition Equation (20) we would find that

$$\left(\sum\nolimits_{k \in \mathcal{K}} \delta_k \boldsymbol{e}_k\right)^\top Q \left(\sum\nolimits_{k \notin \mathcal{K}} \delta_k \boldsymbol{e}_k\right) = \frac{1}{4} (\boldsymbol{y} + \boldsymbol{y}')^\top Q (\boldsymbol{y} - \boldsymbol{y}') = 0.$$

However, X_n is SI-asymmetric and thus there must be $\#\mathcal{K} = 0$ or $\#\mathcal{K} = n$, that is, $\boldsymbol{y}' = -\boldsymbol{y}$ or $\boldsymbol{y}' = \boldsymbol{y}$. Therefore, $d_{\mathcal{H}}(\boldsymbol{h}, \boldsymbol{h}') = 0$ and $\boldsymbol{h}'$ is equivalent to $\boldsymbol{h}$. ■

Appendix B. Proof of Lemma 18

For any $\boldsymbol{h} \in \mathcal{H}'_Q$, there exists $\boldsymbol{\alpha} \in \mathbb{R}^n$ such that $\boldsymbol{h} = U\boldsymbol{\alpha}$, where U consists of n orthonormal eigenvectors of Q, and $\|\boldsymbol{\alpha}\|_2 = 1$ since $\|\boldsymbol{h}\|_2 = 1$ and $U^\top U = I$. The expression $\boldsymbol{h} = U\boldsymbol{\alpha}$ is an unlabeled-labeled representation (ULR) since U only has the information about unlabeled samples extracted from Q. Each column of U has a unit length, and thus $\|U\|_F^2 = n$ where $\|\cdot\|_F$ is the Frobenius norm. The first part of the upper bound, namely,

$$\mathcal{R}_u(\mathcal{H}'_Q) \leq \sqrt{2n/n'(n-n')},$$

comes from Equations (20)–(22) of El-Yaniv and Pechyony (2009).

Let $\hat{Q} = (\gamma Q - \eta^* I_n)^{-1}$. Another ULR is shown in Equation (30), in the proof of Theorem 13:

$$\boldsymbol{h} = \hat{Q}\,\mathrm{sign}(\boldsymbol{h}).$$

It is clear that $1/(\gamma\lambda_1 - \eta^*), \ldots, 1/(\gamma\lambda_n - \eta^*)$ are the eigenvalues of $\hat{Q}$ given that $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of Q. Subsequently, the second part of the upper bound, that is,

$$\mathcal{R}_u(\mathcal{H}'_Q) \leq \sqrt{\frac{2}{n'(n-n')}} \left(\sum_{i=1}^{n} \frac{n}{(\gamma\lambda_i - \eta^*)^2} \right)^{1/2},$$

can be derived from Equations (20)–(22) of El-Yaniv and Pechyony (2009) with $\mu_1 = \sqrt{n}$. Furthermore, Equation (30) is also a kernel ULR, since $\hat{Q}$ is symmetric positive definite and can be viewed as a kernel matrix. Thereby we can obtain the third part of the upper bound

$$\mathcal{R}_u(\mathcal{H}'_Q) \leq \sqrt{\frac{2}{n'(n-n')}} \left(\sum_{i=1}^{n} \frac{\mu}{\gamma\lambda_i - \eta^*} \right)^{1/2}$$

based on Equations (23)–(25) of El-Yaniv and Pechyony (2009) with $\mu_2 = \sqrt{\mu}$. ■

References

F. Agakov and D. Barber. Kernelized infomax clustering. In *Advances in Neural Information Processing Systems 18 (NIPS)*, 2006.

D. Arthur and S. Vassilvitskii. How slow is the k-means method? In *Proceedings of 22nd ACM Symposium on Computational Geometry (SoCG)*, 2006.

P. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14 (NIPS)*, 2002.

S. Ben-David, D. Pál, and H. U. Simon. Stability of *k*-means clustering. In *Proceedings of 20th Annual Conference on Learning Theory (COLT)*, 2007.

P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta Numerica*, 4:1–51, 1995.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of 5th Annual Workshop on Computational Learning Theory (COLT)*, 1992.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In O. Bousquet, U. von Luxburg, and G. Rätsch, editors, *Advanced Lectures on Machine Learning*, pages 169–207. Springer, 2004.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

T. De Bie and N. Cristianini. Convex methods for transduction. In *Advances in Neural Information Processing Systems 16 (NIPS)*, 2004.

T. De Bie and N. Cristianini. Fast SDP relaxations of graph cut clustering, transduction, and other combinatorial problems. *Journal of Machine Learning Research*, 7:1409–1436, 2006.

C. Ding and X. He. K-means clustering via principal component analysis. In *Proceedings of 21st International Conference on Machine Learning (ICML)*, 2004.

R. El-Yaniv and D. Pechyony. Transductive Rademacher complexity and its applications. *Journal of Artificial Intelligence Research*, 35:193–234, 2009.

R. El-Yaniv, D. Pechyony, and V. Vapnik. Large margin vs. large volume in transductive learning. *Machine Learning*, 72(3):173–188, 2008.

L. Faivishevsky and J. Goldberger. A nonparametric information theoretic clustering algorithm. In *Proceedings of 27th International Conference on Machine Learning (ICML)*, 2010.

M. Girolami. Mercer kernel-based clustering in feature space. *IEEE Transactions on Neural Networks*, 13(3):780–784, 2002.

R. Gomes, A. Krause, and P. Perona. Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems 23 (NIPS)*, 2010.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21 (web page and software). http://cvxr.com/cvx, 2011.

J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, 2001.

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.

Y. Li, I. W. Tsang, J. T. Kwok, and Z.-H. Zhou. Tighter and convex maximum margin clustering. In *Proceedings of 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.

J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

M. Meila and J. Shi. A random walks view of spectral segmentation. In *Proceedings of 8th International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2001.

R. Meir and T. Zhang. Generalization error bounds for Bayesian mixture algorithms. *Journal of Machine Learning Research*, 4:839–860, 2003.

A. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14 (NIPS)*, 2002.

G. Niu, B. Dai, L. Shang, and M. Sugiyama. Maximum volume clustering. In *Proceedings of 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

P. Raghavan and C. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. Technical Report UCB/CSD-85-242, UC Berkeley, 1985.

A. Rakhlin and A. Caponnetto. Stability of k-means clustering. In *Advances in Neural Information Processing Systems 19 (NIPS)*, 2007.

C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27: 379–423 & 623–656, 1948.

H. Sherali and W. Adams. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*. Kluwer Academic Publishers, 1998.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

A. Smola, S.V.N. Vishwanathan, and T. Hofmann. Kernel methods for missing variables. In *Proceedings of 10th International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2005.

L. Song, A. Smola, A. Gretton, and K. Borgwardt. A dependence maximization view of clustering. In *Proceedings of 24th International Conference on Machine Learning (ICML)*, 2007.

M. Sugiyama, M. Yamada, M. Kimura, and H. Hachiya. On information-maximization clustering: Tuning parameter selection and analytic solution. In *Proceedings of 28th International Conference on Machine Learning (ICML)*, 2011.

T. Suzuki, M. Sugiyama, J. Sese, and T. Kanamori. Approximating mutual information by maximum likelihood density ratio estimation. In *JMLR Workshop and Conference Proceedings*, volume 4, pages 5–20, 2008.

T. Suzuki, M. Sugiyama, T. Kanamori, and J. Sese. Mutual information estimation reveals global associations between stimuli and biological processes. *BMC Bioinformatics*, 10(1):S52, 2009.

H. Valizadegan and R. Jin. Generalized maximum margin clustering and unsupervised kernel learning. In *Advances in Neural Information Processing Systems 19 (NIPS)*, 2007.

V. N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer Verlag, 1982.

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

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

U. von Luxburg, O. Bousquet, and M. Belkin. Limits of spectral clustering. In *Advances in Neural Information Processing Systems 17 (NIPS)*, 2005.

U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *Annals of Statistics*, 36(2):555–586, 2008.

U. von Luxburg, R. Williamson, and I. Guyon. Clustering: Science or art? In *JMLR Workshop and Conference Proceedings*, volume 27, pages 65–80, 2012.

F. Wang, B. Zhao, and C. Zhang. Linear time maximum margin clustering. *IEEE Transactions on Neural Networks*, 21(2):319–332, 2010.

L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *Proceedings of 20th National Conference on Artificial Intelligence (AAAI)*, 2005.

L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems 17 (NIPS)*, 2005.

A. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.

L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems 17 (NIPS)*, 2005.

H. Zha, X. He, C. Ding, M. Gu, and H. Simon. Spectral relaxation for k-means clustering. In *Advances in Neural Information Processing Systems 14 (NIPS)*, 2002.

K. Zhang, I. W. Tsang, and J. T. Kwok. Maximum margin clustering made practical. In *Proceedings of 24th International Conference on Machine Learning (ICML)*, 2007.

B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *Proceedings of 25th International Conference on Machine Learning (ICML)*, 2008a.

B. Zhao, F. Wang, and C. Zhang. Efficient maximum margin clustering via cutting plane algorithm. In *Proceedings of 8th SIAM International Conference on Data Mining (SDM)*, 2008b.

Journal of Machine Learning Research 14 (2013) 2689-2728 Submitted 2/12; Revised 1/13; Published 9/13

Sparse/Robust Estimation and Kalman Smoothing with Nonsmooth Log-Concave Densities: Modeling, Computation, and Theory*

Aleksandr Y. Aravkin SARAVKIN@US.IBM.COM
IBM T.J. Watson Research Center
1101 Kitchawan Rd
Yorktown Heights, NY 10598

James V. Burke JVBURKE@UW.EDU
Department of Mathematics
Box 354350
University of Washington
Seattle, WA, 98195-4350 USA

Gianluigi Pillonetto GIAPI@DEI.UNIPD.IT
Department of Information Engineering
Via Gradenigo 6/A
University of Padova
Padova, Italy

Editor: Aapo Hyvarinen

Abstract

We introduce a new class of quadratic support (QS) functions, many of which already play a crucial role in a variety of applications, including machine learning, robust statistical inference, sparsity promotion, and inverse problems such as Kalman smoothing. Well known examples of QS penalties include the ℓ_2, Huber, ℓ_1 and Vapnik losses. We build on a dual representation for QS functions, using it to characterize conditions necessary to interpret these functions as negative logs of true probability densities. This interpretation establishes the foundation for statistical modeling with both known and new QS loss functions, and enables construction of non-smooth multivariate distributions with specified means and variances from simple scalar building blocks.

The main contribution of this paper is a flexible statistical modeling framework for a variety of learning applications, together with a toolbox of efficient numerical methods for estimation. In particular, a broad subclass of QS loss functions known as piecewise linear quadratic (PLQ) penalties has a dual representation that can be exploited to design interior point (IP) methods. IP methods solve nonsmooth optimization problems by working directly with smooth systems of equations characterizing their optimality. We provide several numerical examples, along with a code that can be used to solve general PLQ problems.

The efficiency of the IP approach depends on the structure of particular applications. We consider the class of dynamic inverse problems using Kalman smoothing. This class comprises a wide variety of applications, where the aim is to reconstruct the state of a dynamical system with known process and measurement models starting from noisy output samples. In the classical case, Gaus-

*. The authors would like to thank Bradley M. Bell for insightful discussions and helpful suggestions. The research leading to these results has received funding from the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement no 257462 HYCON2 Network of excellence, by the MIUR FIRB project RBFR12M3AC - Learning meets time: a new computational approach to learning in dynamic systems.

sian errors are assumed both in the process and measurement models for such problems. We show that the extended framework allows arbitrary PLQ densities to be used, and that the proposed IP approach solves the generalized Kalman smoothing problem while maintaining the linear complexity in the size of the time series, just as in the Gaussian case. This extends the computational efficiency of the Mayne-Fraser and Rauch-Tung-Striebel algorithms to a much broader nonsmooth setting, and includes many recently proposed robust and sparse smoothers as special cases.

Keywords: statistical modeling, convex analysis, nonsmooth optimization, robust inference, sparsity optimization, Kalman smoothing, interior point methods

1. Introduction

Consider the classical problem of Bayesian parametric regression (MacKay, 1992; Roweis and Ghahramani, 1999) where the unknown $x \in \mathbb{R}^n$ is a random vector,[1] with a prior distribution specified using a known invertible matrix $G \in \mathbb{R}^{n\times n}$ and known vector $\mu \in \mathbb{R}^n$ via

$$\mu = Gx + w\,, \tag{1}$$

where w is a zero mean vector with covariance Q. Let z denote a linear transformation of x contaminated with additive zero mean measurement noise v with covariance R,

$$z = Hx + v\,, \tag{2}$$

where $H \in \mathbb{R}^{\ell\times n}$ is a known matrix, while v and w are independent. It is well known that the (unconditional) minimum variance linear estimator of x, as a function of z, is the solution to the following optimization problem:

$$\min_x \quad (z-Hx)^{\mathrm{T}}R^{-1}(z-Hx) + (\mu - Gx)^{\mathrm{T}}Q^{-1}(\mu - Gx)\,. \tag{3}$$

As we will show, (3) includes estimation problems arising in discrete-time dynamic linear systems which admit a state space representation (Anderson and Moore, 1979; Brockett, 1970). In this context, x is partitioned into N subvectors $\{x_k\}$, where each x_k represents the hidden system state at time instant k. For known data z, the classical Kalman smoother exploits the special structure of the matrices H, G, Q and R to compute the solution of (3) in $O(N)$ operations (Gelb, 1974). This procedure returns the minimum variance estimate of the state sequence $\{x_k\}$ when the additive noise in the system is assumed to be Gaussian.

In many circumstances, the estimator (3) performs poorly; put another way, quadratic penalization on model deviation is a bad model in many situations. For instance, it is not robust with respect to the presence of outliers in the data (Huber, 1981; Gao, 2008; Aravkin et al., 2011a; Farahmand et al., 2011) and may have difficulties in reconstructing fast system dynamics, for example, jumps in the state values (Ohlsson et al., 2012). In addition, sparsity-promoting regularization is often used in order to extract a small subset from a large measurement or parameter vector which has greatest impact on the predictive capability of the estimate for future data. This sparsity principle permeates many well known techniques in machine learning and signal processing, including feature selection, selective shrinkage, and compressed sensing (Hastie and Tibshirani, 1990; Efron et al., 2004; Donoho, 2006). In these cases, (3) is often replaced by a more general formulation

$$\min_x \quad V\left(Hx - z; R\right) + W\left(Gx - \mu; Q\right) \tag{4}$$

1. All vectors are column vectors, unless otherwise specified.

where the loss V may be the ℓ_2-norm, the Huber penalty (Huber, 1981), Vapnik's ε-insensitive loss, used in support vector regression (Vapnik, 1998; Hastie et al., 2001), or the hinge loss, leading to support vector classifiers (Evgeniou et al., 2000; Pontil and Verri, 1998; Schölkopf et al., 2000). The regularizer W may be the ℓ_2-norm, the ℓ_1-norm, as in the LASSO (Tibshirani, 1996), or a weighted combination of the two, yielding the elastic net procedure (Zou and Hastie, 2005). Many learning algorithms using infinite-dimensional reproducing kernel Hilbert spaces as hypothesis spaces (Aronszajn, 1950; Saitoh, 1988; Cucker and Smale, 2001) boil down to solving finite-dimensional problems of the form (4) by virtue of the representer theorem (Wahba, 1998; Schölkopf et al., 2001).

These robust and sparse approaches can often be interpreted as placing non-Gaussian priors on w (or directly on x) and on the measurement noise v. The Bayesian interpretation of (4) has been extensively studied in the statistical and machine learning literature in recent years, and probabilistic approaches used in the analysis of estimation and learning algorithms have been studied (Mackay, 1994; Tipping, 2001; Wipf et al., 2011). Non-Gaussian model errors and priors leading to a great variety of loss and penalty functions are also reviewed by Palmer et al. (2006) using convex-type representations, and integral-type variational representations related to Gaussian scale mixtures.

In contrast to the above approaches, in the first part of the paper, we consider a wide class of quadratic support (QS) functions and exploit their dual representation. This class of functions generalizes the notion of piecewise linear quadratic (PLQ) penalties (Rockafellar and Wets, 1998). The dual representation is the key to identifying which QS loss functions can be associated with a density, which in turn allows us to interpret the solution to the problem (4) as a MAP estimator when the loss functions V and W come from this subclass of QS penalties. This viewpoint allows statistical modeling using non-smooth penalties, such as the ℓ_1, hinge, Huber and Vapnik losses, which are all PLQ penalties. Identifying a statistical interpretation for this class of problems gives us several advantages, including a systematic constructive approach to prescribe mean and variance parameters for the corresponding model; a property that is particularly important for Kalman smoothing.

In addition, the dual representation provides the foundation for efficient numerical methods in estimation based on interior point optimization technology. In the second part of the paper, we derive the Karush-Kuhn-Tucker (KKT) equations for problem (4), and introduce interior point (IP) methods, which are iterative methods to solve the KKT equations using smooth approximations. This is essentially a smoothing approach to many (non-smooth) robust and sparse problems of interest to practitioners. Furthermore, we provide conditions under which the IP methods solve (4) when V and W come from PLQ densities, and describe implementation details for the entire class.

A concerted research effort has recently focused on the solution of regularized large-scale inverse and learning problems, where computational costs and memory limitations are critical. This class of problems includes the popular kernel-based methods (Rasmussen and Williams, 2006; Schölkopf and Smola, 2001; Smola and Schölkopf, 2003), coordinate descent methods (Tseng and Yun, 2008; Lucidi et al., 2007; Dinuzzo, 2011) and decomposition techniques (Joachims, 1998; Lin, 2001; Lucidi et al., 2007), one of which is the widely used sequential minimal optimization algorithm for support vector machines (Platt, 1998). Other techniques are based on kernel approximations, for example, using incomplete Cholesky factorization (Fine and Scheinberg, 2001), approximate eigen-decomposition (Zhang and Kwok, 2010) or truncated spectral representations (Pillonetto and Bell, 2007). Efficient interior point methods have been developed for ℓ_1-regularized problems (Kim et al., 2007), and for support vector machines (Ferris and Munson, 2003).

In contrast, general and efficient solvers for state space estimation problems of the form (4) are missing in the literature. The last part of this paper provides a contribution to fill this gap, spe-

cializing the general results to the dynamic case, and recovering the classical efficiency results of the least-squares formulation. In particular, we design new Kalman smoothers tailored for systems subject to noises coming from PLQ densities. Amazingly, it turns out that the IP method studied by Aravkin et al. (2011a) generalizes perfectly to the entire class of PLQ densities under a simple verifiable non-degeneracy condition. In practice, IP methods converge in a small number of iterations, and the effort per iteration depends on the structure of the underlying problem. We show that the IP iterations for all PLQ Kalman smoothing problems can be computed with a number of operations that scales linearly in N, as in the quadratic case. This theoretical foundation generalizes the results recently obtained by Aravkin et al. (2011a), Aravkin et al. (2011b), Farahmand et al. (2011) and Ohlsson et al. (2012), framing them as particular cases of the general framework presented here.

The paper is organized as follows. In Section 2 we introduce the class of QS convex functions, and give sufficient conditions that allow us to interpret these functions as the negative logs of associated probability densities. In Section 3 we show how to construct QS penalties and densities having a desired structure from basic components, and in particular how multivariate densities can be endowed with prescribed means and variances using scalar building blocks. To illustrates this procedure, further details are provided for the Huber and Vapnik penalties. In Section 4, we focus on PLQ penalties, derive the associated KKT system, and present a theorem that guarantees convergence of IP methods under appropriate hypotheses. In Section 5, we present a few simple well-known problems, and compare a basic IP implementation for these problems with an ADMM implementation (all code is available online). In Section 6, we present the Kalman smoothing dynamic model, formulate Kalman smoothing with PLQ penalties, present the KKT system for the dynamic case, and show that IP iterations for PLQ smoothing preserve the classical computational efficiency known for the Gaussian case. We present numerical examples using both simulated and real data in Section 7, and make some concluding remarks in Section 8. Section A serves as an appendix where supporting mathematical results and proofs are presented.

2. Quadratic Support Functions And Densities

In this section, we introduce the class of Quadratic Support (QS) functions, characterize some of their properties, and show that many commonly used penalties fall into this class. We also give a statistical interpretation to QS penalties by interpreting them as negative log likelihoods of probability densities; this relationship allows prescribing means and variances along with the general quality of the error model, an essential requirement of the Kalman smoothing framework and many other areas.

2.1 Preliminaries

We recall a few definitions from convex analysis, required to specify the domains of QS penalties. The reader is referred to Rockafellar (1970) and Rockafellar and Wets (1998) for more detailed reading.

- (Affine hull) Define the affine hull of any set $C \subset \mathbb{R}^n$, denoted by $\mathrm{aff}(C)$, as the smallest affine set (translated subspace) that contains C.
- (Cone) For any set $C \subset \mathbb{R}^n$, denote by cone C the set $\{tr | r \in C, t \in \mathbb{R}_+\}$.
- (Domain) For $f(x) : \mathbb{R}^n \to \overline{\mathbb{R}} = \{\mathbb{R} \cup \infty\}$, $\mathrm{dom}(f) = \{x : f(x) < \infty\}$.

- (Polars of convex sets) For any convex set $C \subset \mathbb{R}^m$, the polar of C is defined to be

$$C^\circ := \{r | \langle r, d\rangle \leq 1 \; \forall \, d \in C\},$$

  and if C is a convex cone, this representation is equivalent to

$$C^\circ := \{r | \langle r, d\rangle \leq 0 \; \forall \, d \in C\}.$$

- (Horizon cone). Let $C \subset \mathbb{R}^n$ be a nonempty convex set. The horizon cone C^∞ is the convex cone of 'unbounded directions' for C, that is, $d \in C^\infty$ if $C + d \subset C$.

- (Barrier cone). The barrier cone of a convex set C is denoted by $\mathrm{bar}(C)$:

$$\mathrm{bar}(C) := \{x^* | \text{for some } \beta \in \mathbb{R}, \langle x, x^*\rangle \leq \beta \;\; \forall x \in C\}\,.$$

- (Support function). The support function for a set C is denoted by $\delta^*(x\,|\,C)$:

$$\delta^*(x\,|\,C) := \sup_{c \in C} \langle x, c\rangle \;\; .$$

2.2 QS Functions And Densities

We now introduce the QS functions and associated densities that are the focus of this paper. We begin with the dual representation, which is crucial to both establishing a statistical interpretation and to the development of a computational framework.

Definition 1 (Quadratic Support functions and penalties) *A QS function is any function* $\rho(U, M, b, B; \cdot) : \mathbb{R}^n \to \overline{\mathbb{R}}$ *having representation*

$$\rho(U, M, b, B; y) = \sup_{u \in U} \left\{ \langle u, b + By\rangle - \tfrac{1}{2}\langle u, Mu\rangle \right\} \;, \tag{5}$$

where $U \subset \mathbb{R}^m$ is a nonempty convex set, $M \in \mathcal{S}^n_+$ the set of real symmetric positive semidefinite matrices, and $b + By$ is an injective affine transformation in y, with $B \in \mathbb{R}^{m \times n}$, so, in particular, $m \leq n$ and $\mathrm{null}(B) = \{0\}$.

When $0 \in U$, we refer to the associated QS function as a penalty, since it is necessarily non-negative.

Remark 2 *When U is polyhedral, $0 \in U$, $b = 0$ and $B = I$, we recover the basic piecewise linear-quadratic penalties characterized in Rockafellar and Wets (1998, Example 11.18).*

Theorem 3 *Let U, M, B, b be as in Definition 1, and set $K = U^\infty \cap \mathrm{null}(M)$. Then*

$$B^{-1}[\mathrm{bar}(U) + \mathrm{Ran}\,(M) - b] \subset \mathrm{dom}[\rho(U, M, B, b; \cdot)] \subset B^{-1}[K^\circ - b]\,,$$

with equality throughout when $\mathrm{bar}(U) + \mathrm{Ran}\,(M)$ is closed, where $\mathrm{bar}(U) = \mathrm{dom}\,(\delta^*(\cdot\,|\,U))$ is the barrier cone of U. In particular, equality always holds when U is polyhedral.

We now show that many commonly used penalties are special cases of QS (and indeed, of the PLQ) class.

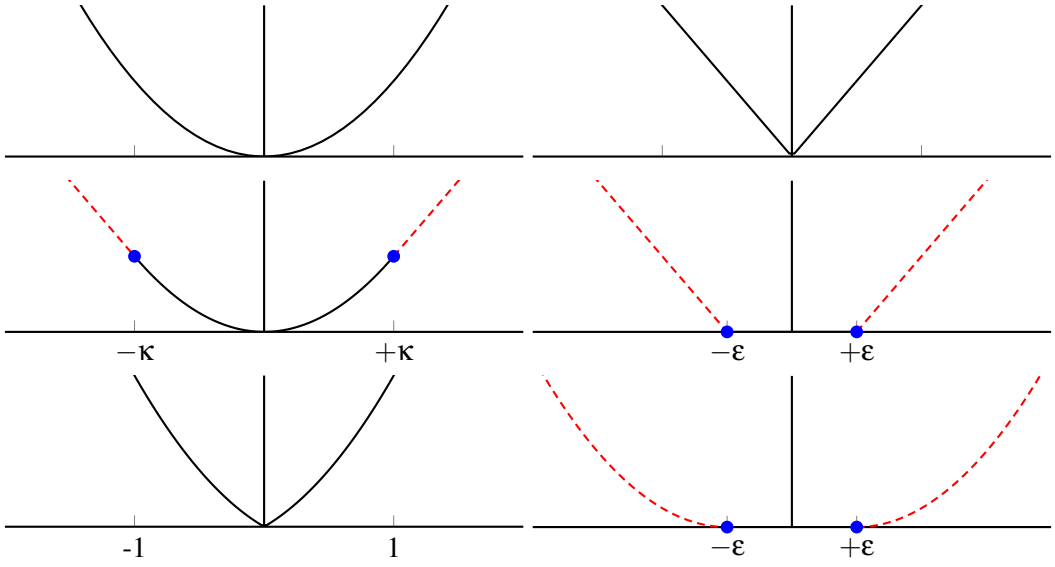


Figure 1: Scalar ℓ_2 (top left), ℓ_1 (top right), Huber (middle left), Vapnik (middle right), elastic net (bottom left) and smooth insensitive loss (bottom right) penalties

Remark 4 (scalar examples) *ℓ_2, ℓ_1, elastic net, Huber, hinge, and Vapnik penalties are all representable using the notation of Definition 1.*

1. *ℓ_2: Take $U = \mathbb{R}$, $M = 1$, $b = 0$, and $B = 1$. We obtain*

$$\rho(y) = \sup_{u \in \mathbb{R}} \left\{ uy - u^2/2 \right\} .$$

The function inside the sup *is maximized at $u = y$, hence $\rho(y) = \frac{1}{2}y^2$, see top left panel of Figure 1.*

2. *ℓ_1: Take $U = [-1,1]$, $M = 0$, $b = 0$, and $B = 1$. We obtain*

$$\rho(y) = \sup_{u \in [-1,1]} \{uy\} .$$

The function inside the sup *is maximized by taking $u = \text{sign}(y)$, hence $\rho(y) = |y|$, see top right panel of Figure 1.*

3. *Elastic net: $\ell_2 + \lambda\ell_1$. Take*

$$U = \mathbb{R} \times [-\lambda, \lambda],\ b = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ M = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} .$$

This construction reveals the general calculus of PLQ addition, see Remark 5. See bottom right panel of Figure 1.

4. *Huber: Take* $U = [-\kappa, \kappa]$, $M = 1$, $b = 0$, *and* $B = 1$. *We obtain*

$$\rho(y) = \sup_{u \in [-\kappa,\kappa]} \left\{ uy - u^2/2 \right\} ,$$

with three explicit cases:

   (a) *If* $y < -\kappa$, *take* $u = -\kappa$ *to obtain* $-\kappa y - \frac{1}{2}\kappa^2$.
   (b) *If* $-\kappa \leq y \leq \kappa$, *take* $u = y$ *to obtain* $\frac{1}{2}y^2$.
   (c) *If* $y > \kappa$, *take* $u = \kappa$ *to obtain a contribution of* $\kappa y - \frac{1}{2}\kappa^2$.

This is the Huber penalty, shown in the middle left panel of Figure 1.

5. *Hinge loss: Taking* $B = 1$, $b = -\varepsilon$, $M = 0$ *and* $U = [0,1]$ *we have*

$$\rho(y) = \sup_{u \in U} \{(y - \varepsilon)u\} = (y - \varepsilon)_+ .$$

To verify this, just note that if $y < \varepsilon$, $u^* = 0$; *otherwise* $u^* = 1$.

6. *Vapnik loss is given by* $(y-\varepsilon)_+ + (-y-\varepsilon)_+$. *We immediately obtain its PLQ representation by taking*

$$B = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \; b = -\begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, \; M = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \; U = [0,1] \times [0,1]$$

to yield

$$\rho(y) = \sup_{u \in U} \left\{ \left\langle \begin{bmatrix} y - \varepsilon \\ -y - \varepsilon \end{bmatrix}, u \right\rangle \right\} = (y-\varepsilon)_+ + (-y-\varepsilon)_+ .$$

The Vapnik penalty is shown in the middle right panel of Figure 1.

7. *Soft hinge loss function (Chu et al., 2001). Combining ideas from examples 4 and 5, we can construct a 'soft' hinge loss; that is, the function*

$$\rho(y) = \begin{cases} 0 & \text{if} \quad y < \varepsilon \\ \frac{1}{2}(y-\varepsilon)^2 & \text{if} \quad \varepsilon < y < \varepsilon + \kappa \\ \kappa(y-\varepsilon) - \frac{1}{2}(\kappa)^2 & \text{if} \quad \varepsilon + \kappa < y \, . \end{cases}$$

that has a smooth (quadratic) transition rather than a kink at ε : *Taking* $B = 1$, $b = -\varepsilon$, $M = 1$ *and* $U = [0, \kappa]$ *we have*

$$\rho(y) = \sup_{u \in [0,\kappa]} \{(y-\varepsilon)u\} - \tfrac{1}{2}u^2 \, .$$

To verify this function has the explicit representation given above, note that if $y < \varepsilon$, $u^* = 0$; *if* $\varepsilon < y < \kappa + \varepsilon$, *we have* $u^* = (y-\varepsilon)_+$, *and if* $\kappa + \varepsilon < y$, *we have* $u^* = \kappa$.

8. *Soft insensitive loss function (Chu et al., 2001). Using example 7, we can create a symmetric soft insensitive loss function (which one might term the Hubnik) by adding together to soft hinge loss functions:*

$$\begin{aligned} \rho(y) &= \sup_{u \in [0,\kappa]} \{(y-\varepsilon)u\} - \tfrac{1}{2}u^2 + \sup_{u \in [0,\kappa]} \{(-y-\varepsilon)u\} - \tfrac{1}{2}u^2 \\ &= \sup_{u \in [0,\kappa]^2} \left\{ \left\langle \begin{bmatrix} y - \varepsilon \\ -y - \varepsilon \end{bmatrix}, u \right\rangle \right\} - \tfrac{1}{2} u^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u \, . \end{aligned}$$

See bottom bottom right panel of Figure 1.

Note that the affine generalization (Definition 1) is needed to form the elastic net, the Vapnik penalty, and the SILF function, as all of these are sums of simpler QS penalties. These sum constructions are examples of a general calculus which allows the modeler to build up a QS density having a desired structure. This calculus is described in the following remark.

Remark 5 *Let $\rho_1(y)$ and $\rho_2(y)$ be two QS penalties specified by U_i, M_i, b_i, B_i, for $i = 1,2$. Then the sum $\rho(y) = \rho_1(y) + \rho_2(y)$ is also a QS penalty, with*

$$U = U_1 \times U_2,\ M = \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix},\ b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},\ B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}\ .$$

Notwithstanding the catalogue of scalar QS functions in Remark 4 and the gluing procedure described in Remark 5, the supremum in Definition 1 appears to be a significant roadblock to understanding and designing a QS function having specific properties. However, with some practice the design of QS penalties is not as daunting a task as it first appears. A key tool in understanding the structure of QS functions are Euclidean norm projections onto convex sets.

Theorem 6 (Projection Theorem for Convex Sets) *[Zarantonello (1971)] Let $Q \in \mathbb{R}^{n\times n}$ be symmetric and positive definite and let $C \subset \mathbb{R}$ be non-empty, closed and convex. Then Q defines an inner product on $\mathbb{R}^n$ by $\langle x, t\rangle_Q = x^T Qy$ with associated Euclidean norm $\|x\|_Q = \sqrt{\langle x, x\rangle_Q}$. The projection of a point $y \in \mathbb{R}^n$ onto C in norm $\|\cdot\|_Q$ is the unique point $P_Q(y \mid C)$ solving the least distance problem*

$$\inf_{x\in C} \|y - x\|_Q, \tag{6}$$

and $z = P_Q(y\,|\,C)$ if and only if $z \in C$ and

$$\langle x - z, y - z\rangle_Q \leq 0 \quad \forall\, x \in C\ . \tag{7}$$

Note that the least distance problem (6) is equivalent to the problem

$$\inf_{x\in C} \tfrac{1}{2}\|y - x\|_Q^2\ .$$

In the following lemma we use projections as well as duality theory to provide alternative representations for QS penalties.

Theorem 7 *Let $M \in \mathbb{R}^{n\times n}$ be symmetric and positive semi-definite matrix, let $L \in \mathbb{R}^{n\times k}$ be any matrix satisfying $M = LL^T$ where $k = \mathrm{rank}(M)$, and let $U \subset \mathbb{R}^n$ be a non-empty, closed and convex set that contains the origin. Then the QS function $\rho := \rho(U, M, 0, I; \cdot)$ has the primal representations*

$$\rho(y) = \inf_{s\in\mathbb{R}^k} \left[\tfrac{1}{2}\|s\|_2^2 + \delta^*(y - Ls\,|\,U)\right] = \inf_{s\in\mathbb{R}^k} \left[\tfrac{1}{2}\|s\|_2^2 + \gamma(y - Ls\,|\,U^\circ)\right]\ , \tag{8}$$

where, for any convex set V,

$$\delta^*(z\,|\,V) := \sup_{v\in V} \langle z, v\rangle \quad \textit{and} \quad \gamma(z\,|\,V) := \inf\{t\,|\,t \geq 0,\ z \in tV\}$$

are the support and gauge functionals for V, respectively.

If it is further assumed that $M \in \mathcal{S}^n_{++}$ the set of positive definite matrices, then ρ has the representations

$$\begin{aligned}
\rho(y) &= \inf_{s\in\mathbb{R}^k} \left[\tfrac{1}{2}\|s\|^2_M + \gamma\left(M^{-1}y - s \,\middle|\, M^{-1}U^\circ\right)\right] && (9)\\
&= \tfrac{1}{2}\|P_M\left(M^{-1}y \,|\, U\right)\|^2_M + \gamma\left(M^{-1}y - P_M(M^{-1}y|\,U) \,\middle|\, M^{-1}U^\circ\right) && (10)\\
&= \inf_{s\in\mathbb{R}^k} \left[\tfrac{1}{2}\|s\|^2_{M^{-1}} + \gamma(y - s \,|\, U^\circ)\right] && (11)\\
&= \tfrac{1}{2}\|P_{M^{-1}}(y|\,MU)\|^2_{M^{-1}} + \gamma(y - P_{M^{-1}}(y|\,MU)\,|\,U^\circ) && (12)\\
&= \tfrac{1}{2}y^TM^{-1}y - \inf_{u\in U} \tfrac{1}{2}\|u - M^{-1}y\|^2_M && (13)\\
&= \tfrac{1}{2}\|P_M(M^{-1}y|\,U)\|^2_M + \left\langle M^{-1}y - P_M(M^{-1}y|\,U),\, P_M(M^{-1}y|\,U)\right\rangle_M && (14)\\
&= \tfrac{1}{2}y^TM^{-1}y - \inf_{v\in MU} \tfrac{1}{2}\|v - y\|^2_{M^{-1}} && (15)\\
&= \tfrac{1}{2}\|P_{M^{-1}}(y|\,MU)\|^2_{M^{-1}} + \langle y - P_{M^{-1}}(y|\,MU),\, P_{M^{-1}}(y|\,MU)\rangle_{M^{-1}}\ . && (16)
\end{aligned}$$

In particular, (15) *says $\rho(y) = \frac{1}{2}y^TM^{-1}y$ whenever $y \in MU$. Also note that, by* (8)*, one can replace the gauge functionals in* (9)-(12) *by the support functional of the appropriate set where $M^{-1}U^\circ = (MU)^\circ$.*

The formulas (9)-(16) show how one can build PLQ penalties having a wide range of desirable properties. We now give a short list of a few examples illustrating how to make use of these representations.

Remark 8 (General examples) *In this remark we show how the representations in Lemma 7 can be used to build QS penalties with specific structure. In each example we specify the components U, M, b, and B for the QS function $\rho := \rho(U, M, b, B; \cdot)$.*

1. *Norms. Any norm $\|\cdot\|$ can be represented as a QS function by taking $M = 0$, $B = I$, $b = 0$, $U = \mathbb{B}^\circ$, where $\mathbb{B}$ is the unit ball of the desired norm. Then, by* (8)*, $\rho(y) = \|y\| = \gamma(y\,|\,\mathbb{B})$.*

2. *Gauges and support functions. Let U be any closed convex set containing the origin, and Take $M = 0, B = I, b = 0$. Then, by* (8)*, $\rho(y) = \gamma(y\,|\,U^\circ) = \delta^*(y\,|\,U)$.*

3. *Generalized Huber functions. Take any norm $\|\cdot\|$ having closed unit ball $\mathbb{B}$. Let $M \in \mathcal{S}^n_{++}$, $B = I$, $b = 0$, and $U = \mathbb{B}^\circ$. Then, by the representation* (12)*,*

$$\rho(y) = \tfrac{1}{2}P_{M^{-1}}(y|\,M\mathbb{B}^\circ)^TM^{-1}P_{M^{-1}}(y|\,M\mathbb{B}^\circ) + \|y - P_{M^{-1}}(y|\,M\mathbb{B}^\circ)\|\ .$$

   In particular, for $y \in M\mathbb{B}^\circ$, $\rho(y) = \frac{1}{2}y^TM^{-1}y$.

   If we take $M = I$ and $\|\cdot\| = \kappa^{-1}\|\cdot\|_1$ for $\kappa > 0$ (i.e., $U = \kappa\mathbb{B}_\infty$ and $U^\circ = \kappa^{-1}\mathbb{B}_1$), then ρ is the multivariate Huber function described in item 4 of Remark 4. In this way, Theorem 7 shows how to generalize the essence of the Huber norm to any choice of norm. For example, if we take $U = \kappa\mathbb{B}_M = \{\kappa u \,|\, \|u\|_M \le 1\}$, then, by (14)*,*

$$\rho(y) = \begin{cases} \frac{1}{2}\|y\|^2_{M^{-1}} & ,\text{if } \|y\|_{M^{-1}} \le \kappa \\ \kappa\|y\|_{M^{-1}} - \frac{\kappa^2}{2} & ,\text{if } \|y\|_{M^{-1}} > \kappa\ . \end{cases}$$

4. *Generalized hinge-loss functions. Let $\|\cdot\|$ be a norm with closed unit ball $\mathbb{B}$, let K be a non-empty closed convex cone in $\mathbb{R}^n$, and let $v \in \mathbb{R}^n$. Set $M = 0$, $b = -v$, $B = I$, and $U = -(\mathbb{B}^\circ \cap K^\circ) = \mathbb{B}^\circ \cap (-K)^\circ$. Then, by (Burke, 1987, Section 2),*

$$\rho(y) = \operatorname{dist}(y \,|\, v - K) = \inf_{u \in K} \|y - b + u\| .$$

If we consider the order structure "$\leq_K$" induced on $\mathbb{R}^n$ by

$$y \leq_K v \quad \Longleftrightarrow \quad v - y \in K ,$$

then $\rho(y) = 0$ if and only if $y \leq_K v$. By taking $\|\cdot\| = \|\cdot\|_1$, $K = \mathbb{R}^n_+$ so $(-K)^\circ = K$, and $v = \varepsilon\mathbf{1}$, where $\mathbf{1}$ is the vector of all ones, we recover the multivariate hinge loss function in Remark 4.

5. *Order intervals and Vapnik loss functions. Let $\|\cdot\|$ be a norm with closed unit ball $\mathbb{B}$, let $K \subset \mathbb{R}^n$ be a non-empty symmetric convex cone in the sense that $K^\circ = -K$, and let $w <_K v$, or equivalently, $v - w \in \operatorname{intr}(K)$. Set*

$$U = (\mathbb{B}^\circ \cap K) \times (\mathbb{B}^\circ \cap K^\circ), \quad M = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad b = -\begin{pmatrix} v \\ w \end{pmatrix}, \quad \text{and} \quad B = \begin{bmatrix} I \\ I \end{bmatrix} .$$

Then

$$\rho(y) = \operatorname{dist}(y \,|\, v - K) + \operatorname{dist}(y \,|\, w + K) .$$

Observe that $\rho(y) = 0$ if and only if $w \leq_K y \leq_K v$. The set $\{y \,|\, w \leq_K y \leq_K v\}$ is an "order interval" (Schaefer, 1970). If we take $w = -v$, then $\{y \,|\, -v \leq_K y \leq_K v\}$ is a symmetric neighborhood of the origin. By taking $\|\cdot\| = \|\cdot\|_1$, $K = \mathbb{R}^n_+$, and $v = \varepsilon\mathbf{1}$=-$w$, we recover the multivariate Vapnik loss function in Remark 4. Further examples of symmetric cones are $\mathcal{S}^n_+$ and the Lorentz or ℓ_2 cone (Güler and Hauser, 2002).

The examples given above show that one can also construct generalized versions of the elastic net as well as the soft insensitive loss functions defined in Remark 4. In addition, cone constraints can also be added by using the identity $\delta^*(\cdot \,|\, K^\circ) = \delta(\cdot \,|\, K)$. These examples serve to illustrate the wide variety of penalty functions representable as QS functions. Computationally, one is only limited by the ability to compute projections described in Theorem 7. For more details on computational properties for QS functions, see Aravkin et al. (2013), Section 6.

In order to characterize QS functions as negative logs of density functions, we need to ensure the integrability of said density functions. The function $\rho(y)$ is said to be *coercive* if $\lim_{\|y\| \to \infty} \rho(y) = \infty$, and coercivity turns out to be the key property to ensure integrability. The proof of this fact and the characterization of coercivity for QS functions are the subject of the next two theorems (see Appendix for proofs).

Theorem 9 (QS integrability) *Suppose $\rho(y)$ is a coercive QS penalty. Then the function $\exp[-\rho(y)]$ is integrable on $\operatorname{aff}[\operatorname{dom}(\rho)]$ with respect to the $\dim(\operatorname{aff}[\operatorname{dom}(\rho)])$-dimensional Lebesgue measure.*

Theorem 10 *A QS function ρ is coercive if and only if $[B^{\mathrm{T}}\operatorname{cone}(U)]^\circ = \{0\}$.*

Theorem 10 can be used to show the coercivity of familiar penalties. In particular, note that if $B = I$, then the QS function is coercive if and only if U contains the origin in its interior.

Corollary 11 *The penalties ℓ_2, ℓ_1, elastic net, Vapnik, and Huber are all coercive.*

Proof We show that all of these penalties satisfy the hypothesis of Theorem 10.

ℓ_2: $U = \mathbb{R}$ and $B = 1$, so $\left[B^{\mathrm{T}}\mathrm{cone}(U)\right]^\circ = \mathbb{R}^\circ = \{0\}$.

ℓ_1: $U = [-1,1]$, so $\mathrm{cone}(U) = \mathbb{R}$, and $B = 1$.

Elastic Net: In this case, $\mathrm{cone}(U) = \mathbb{R}^2$ and $B = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$.

Huber: $U = [-\kappa,\kappa]$, so $\mathrm{cone}(U) = \mathbb{R}$, and $B = 1$.

Vapnik: $U = [0,1]\times[0,1]$, so $\mathrm{cone}(U) = \mathbb{R}^2_+$. $B = \left[\begin{smallmatrix}1\\-1\end{smallmatrix}\right]$, so $B^{\mathrm{T}}\mathrm{cone}(U) = \mathbb{R}$.

■

One can also show the coercivity of the above examples using their primal representations. However, our main objective is to pave the way for a modeling framework where multi-dimensional penalties can be constructed from simple building blocks and then solved by a uniform approach using the dual representations alone.

We now define a family of distributions on $\mathbb{R}^n$ by interpreting piecewise linear quadratic functions ρ as negative logs of corresponding densities. Note that the support of the distributions is always contained in $\mathrm{dom}\,\rho$, which is characterized in Theorem 3.

Definition 12 (QS densities) *Let $\rho(U,M,B,b;y)$ be any coercive extended QS penalty on $\mathbb{R}^n$. Define $\mathbf{p}(y)$ to be the following density on $\mathbb{R}^n$:*

$$\mathbf{p}(y) = \begin{cases} c^{-1}\exp\left[-\rho(y)\right] & y \in \mathrm{dom}\,\rho \\ 0 & \text{else,} \end{cases}$$

where

$$c = \left(\int_{y\in\mathrm{dom}\,\rho} \exp\left[-\rho(y)\right] dy\right),$$

and the integral is with respect to the $\dim(\mathrm{dom}(\rho))$*-dimensional Lebesgue measure.*

QS densities are true densities on the affine hull of the domain of ρ. The proof of Theorem 9 can be easily adapted to show that they have moments of all orders.

3. Constructing QS Densities

In this section, we describe how to construct multivariate QS densities with prescribed means and variances. We show how to compute normalization constants to obtain scalar densities, and then extend to multivariate densities using linear transformations. Finally, we show how to obtain the data structures U,M,B,b corresponding to multivariate densities, since these are used by the optimization approach in Section 4.

We make use of the following definitions. Given a sequence of column vectors $\{r_k\} = \{r_1,\ldots,r_N\}$ and matrices $\{\Sigma_k\} = \{\Sigma_1,\ldots,\Sigma_N\}$, we use the notation

$$\mathrm{vec}(\{r_k\}) = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{bmatrix}, \ \mathrm{diag}(\{\Sigma_k\}) = \begin{bmatrix} \Sigma_1 & 0 & \cdots & 0 \\ 0 & \Sigma_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \Sigma_N \end{bmatrix}.$$

In definition 12, QS densities are defined over $\mathbb{R}^n$. The moments of these densities depend in a nontrivial way on the choice of parameters b, B, U, M. In practice, we would like to be able to construct these densities to have prescribed means and variances. We show how this can be done using scalar QS random variables as the building blocks. Suppose $y = \text{vec}(\{y_k\})$ is a vector of independent (but not necessarily identical) QS random variables with mean 0 and variance 1. Denote by b_k, B_k, U_k, M_k the specification for the densities of y_k. To obtain the density of y, we need only take

$$\begin{aligned} U &= U_1 \times U_2 \times \cdots \times U_N, \\ M &= \text{diag}(\{M_k\}), \\ B &= \text{diag}(\{B_k\}), \\ b &= \text{vec}(\{b_k\}) \, . \end{aligned}$$

For example, the standard Gaussian distribution is specified by $U = \mathbb{R}^n$, $M = I$, $b = 0$, $B = I$, while the standard ℓ_1-Laplace (Aravkin et al., 2011a) is specified by $U = [-1,1]^n$, $M = 0$, $b = 0$, $B = \sqrt{2}I$.

The random vector $\tilde{y} = Q^{1/2}(y + \mu)$ has mean μ and variance Q. If c is the normalizing constant for the density of y, then $c \det(Q)^{1/2}$ is the normalizing constant for the density of $\tilde{y}$.

Remark 13 *Note that only independence of the building blocks is required in the above result. This allows the flexibility to impose different QS densities on different errors in the model. Such flexibility may be useful for example when combining measurement data from different instruments, where some instruments may occasionally give bad data (with outliers), while others have errors that are modeled well by Gaussian distributions.*

We now show how to construct scalar building blocks with mean 0 and variance 1, that is, how to compute the key normalizing constants for any QS penalty. To this aim, suppose $\rho(y)$ is a scalar QS penalty that is symmetric about 0. We would like to construct a density $\mathbf{p}(y) = \exp\left[-\rho(c_2 y)\right]/c_1$ to be a true density with unit variance, that is,

$$\frac{1}{c_1}\int \exp\left[-\rho(c_2 y)\right] dy = 1 \quad \text{and} \quad \frac{1}{c_1}\int y^2 \exp\left[-\rho(c_2 y)\right] dy = 1, \tag{17}$$

where the integrals are over $\mathbb{R}$. Using u-substitution, these equations become

$$c_1 c_2 = \int \exp\left[-\rho(y)\right] dy \quad \text{and} \quad c_1 c_2^3 = \int y^2 \exp\left[-\rho(y)\right] dy.$$

Solving this system yields

$$\begin{aligned} c_2 &= \sqrt{\int y^2 \exp\left[-\rho(y)\right] dy \Big/ \int \exp\left[-\rho(y)\right] dy}, \\ c_1 &= \frac{1}{c_2}\int \exp\left[-\rho(y)\right] dy \, . \end{aligned}$$

These expressions can be used to obtain the normalizing constants for any particular ρ using simple integrals.

3.1 Huber Density

The scalar density corresponding to the Huber penalty is constructed as follows. Set

$$\mathbf{p_H}(y) = \frac{1}{c_1}\exp[-\rho_H(c_2 y)]\ ,$$

where c_1 and c_2 are chosen as in (17). Specifically, we compute

$$\int \exp\left[-\rho_H(y)\right] dy = 2\exp\left[-\kappa^2/2\right]\frac{1}{\kappa} + \sqrt{2\pi}[2\Phi(\kappa)-1],$$
$$\int y^2 \exp\left[-\rho_H(y)\right] dy = 4\exp\left[-\kappa^2/2\right]\frac{1+\kappa^2}{\kappa^3} + \sqrt{2\pi}[2\Phi(\kappa)-1]\ ,$$

where Φ is the standard normal cumulative density function. The constants c_1 and c_2 can now be readily computed.

To obtain the multivariate Huber density with variance Q and mean μ, let $U = [-\kappa,\kappa]^n$, $M = I$, $B = I$ any full rank matrix, and $b = 0$. This gives the desired density:

$$\mathbf{p_H}(y) = \frac{1}{c_1^n \det(Q^{1/2})}\exp\left[-\sup_{u\in U}\left\{\left\langle c_2 Q^{-1/2}(y-\mu), u\right\rangle - \frac{1}{2}u^{\mathrm{T}}u\right\}\right].$$

3.2 Vapnik Density

The scalar density associated with the Vapnik penalty is constructed as follows. Set

$$\mathbf{p_V}(y) = \frac{1}{c_1}\exp\left[-\rho_V(c_2 y)\right]\ ,$$

where the normalizing constants c_1 and c_2 can be obtained from

$$\int \exp\left[-\rho_V(y)\right] dy = 2(\varepsilon+1),$$
$$\int y^2 \exp\left[-\rho_V(y)\right] dy = \frac{2}{3}\varepsilon^3 + 2(2-2\varepsilon+\varepsilon^2),$$

using the results in Section 3. Taking $U = [0,1]^{2n}$, the multivariate Vapnik distribution with mean μ and variance Q is

$$\mathbf{p_V}(y) = \frac{1}{c_1^n \det(Q^{1/2})}\exp\left[-\sup_{u\in U}\left\{\left\langle c_2 B Q^{-1/2}(y-\mu) - \varepsilon \mathbf{1}_{2n}, u\right\rangle\right\}\right]\ ,$$

where B is block diagonal with each block of the form $B = \left[\begin{smallmatrix}1\\-1\end{smallmatrix}\right]$, and $\mathbf{1}_{2n}$ is a column vector of 1's of length $2n$.

4. Optimization With PLQ Penalties

In the previous sections, QS penalties were characterized using their dual representation and interpreted as negative log likelihoods of true densities. As we have seen, the scope of such densities is extremely broad. Moreover, these densities can easily be constructed to possess specified moment

properties. In this section, we expand on their utility by showing that the resulting estimation problems (4) can be solved with high accuracy using standard techniques from numerical optimization for a large subclass of these penalties. We focus on PLQ penalties for the sake of simplicity in our presentation of an interior point approach to solving these estimation problems. However, the interior point approach applies in much more general settings (Nemirovskii and Nesterov, 1994). Nonetheless, the PLQ case is sufficient to cover all of the examples given in Remark 4 while giving the flavor of how to proceed in the more general cases.

We exploit the dual representation for the class of PLQ penalties (Rockafellar and Wets, 1998) to explicitly construct the Karush-Kuhn-Tucker (KKT) conditions for a wide variety of model problems of the form (4). Working with these systems opens the door to using a wide variety of numerical methods for convex quadratic programming to solve (4).

Let $\rho(U_v, M_v, b_v, B_v; y)$ and $\rho(U_w, M_w, b_w, B_w; y)$ be two PLQ penalties and define

$$V(v;R) := \rho(U_v, M_v, b_v, B_v; R^{-1/2}v) \tag{18}$$

and

$$W(w;Q) := \rho(U_w, M_w, b_w, B_w; Q^{-1/2}w). \tag{19}$$

Then (4) becomes

$$\min_{y\in\mathbb{R}^n} \rho(U, M, b, B; y), \tag{20}$$

where

$$U := U_v \times U_w,\ M := \begin{bmatrix} M_v & 0 \\ 0 & M_w \end{bmatrix},\ b := \begin{pmatrix} b_v - B_v R^{-1/2} z \\ b_w - B_w Q^{-1/2}\mu \end{pmatrix},$$

and

$$B := \begin{bmatrix} B_v R^{-1/2} H \\ B_w Q^{-1/2} G \end{bmatrix}.$$

Moreover, the hypotheses in (1), (2), (4), and (5) imply that the matrix B in (20) is injective. Indeed, $By = 0$ if and only if $B_w Q^{-1/2} G y = 0$, but, since G is nonsingular and B_w is injective, this implies that $y = 0$. That is, $\mathrm{nul}(B) = \{0\}$. Consequently, the objective in (20) takes the form of a PLQ penalty function (5). In particular, if (18) and (19) arise from PLQ densities (definition 12), then the solution to problem (20) is the MAP estimator in the statistical model (1)-(2).

To simplify the notational burden, in the remainder of this section we work with (20) directly and assume that the defining objects in (20) have the dimensions specified in (5);

$$U \in \mathbb{R}^m,\ M \in \mathbb{R}^{m\times m},\ b \in \mathbb{R}^m, \text{ and } B \in \mathbb{R}^{m\times n}.$$

The Lagrangian (Rockafellar and Wets, 1998)[Example 11.47] for problem (20) is given by

$$L(y,u) = b^{\mathrm{T}}u - \frac{1}{2}u^{\mathrm{T}}Mu + u^{\mathrm{T}}By\,.$$

By assumption U is polyhedral, and so can be specified to take the form

$$U = \{u : A^{\mathrm{T}}u \le a\}\,, \tag{21}$$

where $A \in \mathbb{R}^{m\times\ell}$. Using this representation for U, the optimality conditions for (20) (Rockafellar, 1970; Rockafellar and Wets, 1998) are

$$\begin{aligned} 0 &= B^{\mathrm{T}}u, \\ 0 &= b + By - Mu - Aq, \\ 0 &= A^{\mathrm{T}}u + s - a, \\ 0 &= q_i s_i \ , \ i = 1,\ldots,\ell \ , \ q,s \geq 0 \ , \end{aligned} \tag{22}$$

where the non-negative slack variable s is defined by the third equation in (22). The non-negativity of s implies that $u \in U$. The equations $0 = q_i s_i \ , \ i = 1,\ldots,\ell$ in (22) are known as the complementarity conditions. By convexity, solving the problem (20) is equivalent to satisfying (22). There is a vast optimization literature on working directly with the KKT system. In particular, interior point (IP) methods (Kojima et al., 1991; Nemirovskii and Nesterov, 1994; Wright, 1997) can be employed. In the Kalman filtering/smoothing application, IP methods have been used to solve the KKT system (22) in a numerically stable and efficient manner, see, for example, Aravkin et al. (2011b). Remarkably, the IP approach studied by Aravkin et al. (2011b) generalizes to the entire PLQ class. For Kalman filtering and smoothing, the computational efficiency is also preserved (see Section 6). Here, we show the general development for the entire PLQ class using standard techniques from the IP literature, see, for example, Kojima et al. (1991).

Let U, M, b, B, and A be as defined in (5) and (21), and let $\tau \in (0, +\infty]$. We define the τ *slice of the strict feasibility region for* (22) to be the set

$$\mathcal{F}_+(\tau) = \left\{ (s,q,u,y) \;\middle|\; \begin{array}{c} 0 < s,\ 0 < q,\ s^{\mathrm{T}}q \leq \tau, \text{ and} \\ (s,q,u,y) \text{ satisfy the affine equations in (22)} \end{array} \right\},$$

and the *central path for* (22) to be the set

$$\mathcal{C} := \left\{ (s,q,u,y) \;\middle|\; \begin{array}{c} 0 < s,\ 0 < q,\ \gamma = q_i s_i \ \ i = 1,\ldots,\ell, \text{ and} \\ (s,q,u,y) \text{ satisfy the affine equations in (22)} \end{array} \right\}.$$

For simplicity, we define $\mathcal{F}_+ := \mathcal{F}_+(+\infty)$. The basic strategy of a primal-dual IP method is to follow the central path to a solution of (22) as $\gamma \downarrow 0$ by applying a predictor-corrector damped Newton method to the function mapping $\mathbb{R}^\ell \times \mathbb{R}^\ell \times \mathbb{R}^m \times \mathbb{R}^n$ to itself given by

$$F_\gamma(s,q,u,y) = \begin{bmatrix} s + A^{\mathrm{T}}u - a \\ D(q)D(s)\mathbf{1} - \gamma\mathbf{1} \\ By - Mu - Aq + b \\ B^{\mathrm{T}}u \end{bmatrix},$$

where $D(q)$ and $D(s)$ are diagonal matrices with vectors q, s on the diagonal.

Theorem 14 *Let U, M, b, B, and A be as defined in* (5) *and* (21). *Given $\tau > 0$, let $\mathcal{F}_+$, $\mathcal{F}_+(\tau)$, and $\mathcal{C}$ be as defined above. If*

$$\mathcal{F}_+ \neq \emptyset \quad \textit{and} \quad \mathrm{null}(M) \cap \mathrm{null}(A^{\mathrm{T}}) = \{0\}, \tag{23}$$

then the following statements hold.

(i) $F_\gamma^{(1)}(s,q,u,y)$ is invertible for all $(s,q,u,y) \in \mathcal{F}_+$.

(ii) *Define* $\widehat{\mathcal{F}}_+ = \{(s,q) \,|\, \exists (u,y) \in \mathbb{R}^m \times \mathbb{R}^n \text{ s.t. } (s,q,u,y) \in \mathcal{F}_+ \}$. *Then for each* $(s,q) \in \widehat{\mathcal{F}}_+$ *there exists a unique* $(u,y) \in \mathbb{R}^m \times \mathbb{R}^n$ *such that* $(s,q,u,y) \in \mathcal{F}_+$.

(iii) *The set* $\mathcal{F}_+(\tau)$ *is bounded for every* $\tau > 0$.

(iv) *For every* $g \in \mathbb{R}^{\ell}_{++}$, *there is a unique* $(s,q,u,y) \in \mathcal{F}_+$ *such that* $g = (s_1 q_1, s_2 q_2, \ldots, s_\ell q_\ell)^{\mathrm{T}}$.

(v) *For every* $\gamma > 0$, *there is a unique solution* $[s(\gamma), q(\gamma), u(\gamma), y(\gamma)]$ *to the equation* $F_\gamma(s,q,u,y) = 0$. *Moreover, these points form a differentiable trajectory in* $\mathbb{R}^{\nu} \times \mathbb{R}^{\nu} \times \mathbb{R}^m \times \mathbb{R}^n$. *In particular, we may write*

$$\mathcal{C} = \{ [s(\gamma), q(\gamma), u(\gamma), y(\gamma)] \,|\, \gamma > 0 \} \ .$$

(vi) *The set of cluster points of the central path as* $\gamma \downarrow 0$ *is non-empty, and every such cluster point is a solution to* (22).

Please see the Appendix for proof. Theorem 14 shows that if the conditions (23) hold, then IP techniques can be applied to solve the problem (20). In all of the applications we consider, the condition $\mathrm{null}(M) \cap \mathrm{null}(A^{\mathrm{T}}) = \{0\}$ is easily verified. For example, in the setting of (20) with

$$U_v = \{u \,|\, A_v u \leq a_v\} \quad \text{and} \quad U_w = \{u \,|\, A_w u \leq b_w\} \tag{24}$$

this condition reduces to

$$\mathrm{null}(M_v) \cap \mathrm{null}(A_v^{\mathrm{T}}) = \{0\} \quad \text{and} \quad \mathrm{null}(M_w) \cap \mathrm{null}(A_w^{\mathrm{T}}) = \{0\}. \tag{25}$$

Corollary 15 *The densities corresponding to* ℓ_1, ℓ_2, *Huber, and Vapnik penalties all satisfy hypothesis* (25).

Proof We verify that $\mathrm{null}(M) \cap \mathrm{null}(A^{\mathrm{T}}) = 0$ for each of the four penalties. In the ℓ_2 case, M has full rank. For the ℓ_1, Huber, and Vapnik penalties, the respective sets U are bounded, so $U^{\infty} = \{0\}$. ■

On the other hand, the condition $\mathcal{F}_+ \neq \emptyset$ is typically more difficult to verify. We show how this is done for two sample cases from class (4), where the non-emptiness of $\mathcal{F}_+$ is established by constructing an element of this set. Such constructed points are useful for initializing the interior point algorithm.

4.1 ℓ_1-ℓ_2

Suppose $V(v;R) = \left\|R^{-1/2} v\right\|_1$ and $W(w;Q) = \frac{1}{2}\left\|Q^{-1/2} w\right\|_2^2$. In this case

$$U_v = [-\mathbf{1}_m, \mathbf{1}_m],\ M_v = 0_{m\times m},\ b_v = 0_m,\ B_v = I_{m\times m},$$
$$U_w = \mathbb{R}^n,\ M_w = I_{n\times n},\ b_w = 0_n,\ B_w = I_{n\times n},$$

and $R \in \mathbb{R}^{m\times m}$ and $Q \in \mathbb{R}^{n\times n}$ are symmetric positive definite covariance matrices. Following the notation of (20) we have

$$U = [-\mathbf{1}, \mathbf{1}] \times \mathbb{R}^n,\ M = \begin{bmatrix} 0_{m\times m} & 0 \\ 0 & I_{n\times n} \end{bmatrix},\ b = \begin{pmatrix} -R^{-1/2} z \\ -Q^{-1/2}\mu \end{pmatrix},\ B = \begin{bmatrix} R^{-1/2} H \\ Q^{-1/2} G \end{bmatrix}.$$

The specification of U in (21) is given by

$$A^{\mathrm{T}} = \begin{bmatrix} I_{m\times m} & 0_{n\times n} \\ -I_{m\times m} & 0_{n\times n} \end{bmatrix} \text{ and } a = \begin{pmatrix} \mathbf{1} \\ -\mathbf{1} \end{pmatrix} .$$

Clearly, the condition $\mathrm{null}(M)\cap\mathrm{null}(A^{\mathrm{T}}) = \{0\}$ in (23) is satisfied. Hence, for Theorem 14 to apply, we need only check that $\mathcal{F}_+ \neq \emptyset$. This is easily established by noting that $(s,q,u,y) \in \mathcal{F}_+$, where

$$u = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ y = G^{-1}\mu,\ s = \begin{pmatrix} \mathbf{1} \\ \mathbf{1} \end{pmatrix},\ q = \begin{pmatrix} \mathbf{1} + [R^{-1/2}(Hy - z)]_+ \\ \mathbf{1} - [R^{-1/2}(Hy - z)]_- \end{pmatrix},$$

where, for $g \in \mathbb{R}^\ell$, g_+ is defined componentwise by $g_{+(i)} = \max\{g_i, 0\}$ and $g_{-(i)} = \min\{g_i, 0\}$.

4.2 Vapnik-Huber

Suppose that $V(v;R)$ and $W(w;Q)$ are as in (18) and (19), respectively, with V a Vapnik penalty and W a Huber penalty:

$$U_v = [0, \mathbf{1}_m] \times [0, \mathbf{1}_m],\ M_v = 0_{2m\times 2m},\ b_v = -\begin{pmatrix} \varepsilon\mathbf{1}_m \\ \varepsilon\mathbf{1}_m \end{pmatrix},\ B_v = \begin{bmatrix} I_{m\times m} \\ -I_{m\times m} \end{bmatrix},$$
$$U_w = [-\kappa\mathbf{1}_n, \kappa\mathbf{1}_n],\ M_w = I_{n\times n},\ b_w = 0_n,\ B_w = I_{n\times n} ,$$

and $R \in \mathbb{R}^{m\times m}$ and $Q \in \mathbb{R}^{n\times n}$ are symmetric positive definite covariance matrices. Following the notation of (20) we have

$$U = ([0, \mathbf{1}_m] \times [0, \mathbf{1}_m]) \times [-\kappa\mathbf{1}_n, \kappa\mathbf{1}_n],\ M = \begin{bmatrix} 0_{2m\times 2m} & 0 \\ 0 & I_{n\times n} \end{bmatrix},$$
$$b = -\begin{pmatrix} \varepsilon\mathbf{1}_m + R^{-1/2}z \\ \varepsilon\mathbf{1}_m - R^{-1/2}z \\ Q^{-1/2}\mu \end{pmatrix},\ B = \begin{bmatrix} R^{-1/2}H \\ -R^{-1/2}H \\ Q^{-1/2}G \end{bmatrix} .$$

The specification of U in (21) is given by

$$A^{\mathrm{T}} = \begin{bmatrix} I_{m\times m} & 0 & 0 \\ -I_{m\times m} & 0 & 0 \\ 0 & I_{m\times m} & 0 \\ 0 & -I_{m\times m} & 0 \\ 0 & 0 & I_{n\times n} \\ 0 & 0 & -I_{n\times n} \end{bmatrix} \text{ and } a = \begin{pmatrix} \mathbf{1}_m \\ 0_m \\ \mathbf{1}_m \\ 0_m \\ \kappa\mathbf{1}_n \\ \kappa\mathbf{1}_n \end{pmatrix} .$$

Since $\mathrm{null}(A^{\mathrm{T}}) = \{0\}$, the condition $\mathrm{null}(M)\cap\mathrm{null}(A^{\mathrm{T}}) = \{0\}$ in (23) is satisfied. Hence, for Theorem 14 to apply, we need only check that $\mathcal{F}_+ \neq \emptyset$. We establish this by constructing an element (s,q,u,y) of $\mathcal{F}_+$. For this, let

$$u = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix},\ s = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{pmatrix},\ q = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \end{pmatrix},$$

and set

$$y = 0_n,\ u_1 = u_2 = \tfrac{1}{2}\mathbf{1}_\ell,\ u_3 = 0_n,\ s_1 = s_2 = s_3 = s_4 = \tfrac{1}{2}\mathbf{1}_\ell,\ s_5 = s_6 = \kappa\mathbf{1}_n,$$

and

$$\begin{aligned} q_1 &= \mathbf{1}_m - (\varepsilon\mathbf{1}_m + R^{-1/2}z)_-,\ q_2 = \mathbf{1}_m + (\varepsilon\mathbf{1}_m + R^{-1/2}z)_+, \\ q_3 &= \mathbf{1}_m - (\varepsilon\mathbf{1}_m - R^{-1/2}z)_-,\ q_4 = \mathbf{1}_m + (\varepsilon\mathbf{1}_m - R^{-1/2}z)_+, \\ q_5 &= \mathbf{1}_n - (Q^{-1/2}\mu)_-,\ q_6 = \mathbf{1}_n + (Q^{-1/2}\mu)_+ \,. \end{aligned}$$

Then $(s,q,u,y) \in \mathcal{F}_+$.

5. Simple Numerical Examples And Comparisons

Before we proceed to the main application of interest (Kalman smoothing), we present a few simple and interesting problems in the PLQ class. An IP solver that handles the problems discussed in this section is available through `github.com/saravkin/`, along with example files and ADMM implementations. A comprehensive comparison with other methods is not in our scope, but we do compare the IP framework with the Alternating Direction Method of Multipliers (ADMM), see Boyd et al. (2011) for a tutorial reference. We hope that the examples and the code will help readers to develop intuition about these two methods.

We focus on ADMM in particular because these methods enjoy widespread use in machine learning and other applications, due to their versatility and ability to scale to large problems. The fundamental difference between ADMM and IP is that ADMM methods have at best linear convergence, so they cannot reach high accuracy in reasonable time, see [Section 3.2.2] of Boyd et al. (2011). In contrast, IP methods have a superlinear convergence rate. In fact, some variants have 2-step quadratic convergence, see Ye and Anstreicher (1993) and Wright (1997).

In addition to accuracy concerns, IP methods may be preferable to ADMM when

- objective contains complex non-smooth terms, for example, $\|Ax - b\|_1$.
- linear operators within the objective formulations are ill-conditioned.

For formulations with well-conditioned linear operators and simple nonsmooth pieces (such as Lasso), ADMM can easily outperform IP. In these cases ADMM methods can attain moderate accuracy (and good solutions) very quickly, by exploiting partial smoothness and/or simplicity of regularizing functionals. For problems lacking these features, such as general formulations built from (nonsmooth) PLQ penalties and possibly ill-conditioned linear operators, IP can dominate ADMM, reaching the true solution while ADMM struggles.

We present a few simple examples below, either developing the ADMM approach for each, or discussing the difficulties (when applicable). We explain advantages and disadvantages of using IP, and present numerical results. A simple IP solver that handles all of the examples, together with ADMM code used for the comparisons, is available through `github.com/saravkin/`. The Lasso example was taken directly from `http://www.stanford.edu/~boyd/papers/admm/`, and we implemented the other ADMM examples using this code as a template.

5.1 Lasso Problem

Consider the Lasso problem

$$\min_x \frac{1}{2}\|Ax-b\|_2^2+\lambda\|x\|_1 ,$$

where $A\in\mathbb{R}^{n\times m}$. Assume that $m<n$. In order to develop an ADMM approach, we split the variables and introduce a constraint:

$$\min_{x,z} \frac{1}{2}\|Ax-b\|_2^2+\lambda\|z\|_1 \quad \text{s.t.} \quad x=z . \tag{26}$$

The augmented Lagrangian for (26) is given by

$$\mathcal{L}(x,z,y)=\frac{1}{2}\|Ax-b\|_2^2+\lambda\|z\|_1+\eta y^T(z-x)+\frac{\rho}{2}\|z-x\|_2^2 ,$$

where η is the augmented Lagrangian parameter. The ADMM method now comprises the following iterative updates:

$$\begin{aligned}
x^{k+1} &= \operatorname*{argmin}_x \frac{1}{2}\|Ax-b\|_2^2+\frac{\eta}{2}\|x+y^k-z^k\|_2^2,\\
z^{k+1} &= \operatorname*{argmin}_z \lambda\|z\|_1+\frac{\eta}{2}\|z-x^{k+1}+y^k\|_2^2,\\
y^{k+1} &= y^k+(z^{k+1}-x^{k+1}) .
\end{aligned}$$

Turning our attention to the x-update, note that the gradient is given by

$$A^T(Ax-b)+\eta(x+y^k-z^k)=(A^TA+I)x-A^Tb+\eta(y^k-z^k) .$$

At every iteration, the update requires solving the same positive definite $m\times m$ symmetric system. Forming A^TA+I is $O(nm^2)$ time, and obtaining a Cholesky factorization is $O(m^3)$, but once this is done, every x-update can be obtained in $O(m^2)$ time by doing two back-solves.

The z-update has a closed form solution given by soft thresholding:

$$z^{k+1}=S(x^{k+1}-y^{k+1},\lambda/\eta) ,$$

which is an $O(n)$ operation. The multiplier update is also $O(n)$. Therefore, the complexity per iteration is $O(m^2+n)$, making ADMM a great method for this problem.

In contrast, *each iteration* of IP is dominated by the complexity of forming a dense $m\times m$ system A^TD^kA, where D^k is a diagonal matrix that depends on the iteration. So while both methods require an investment of $O(nm^2)$ to form and $O(m^3)$ to factorize the system, ADMM requires this only at the outset, while IP has to repeat the computation for every iteration. A simple test shows ADMM can find a good answer, with a significant speed advantage already evident for moderate (1000×5000) well-conditioned systems (see Table 1).

5.2 Linear Support Vector Machines

The support vector machine problem can be formulated as the PLQ (see Ferris and Munson, 2003, Section 2.1)

$$\min_{w,\gamma} \frac{1}{2}\|w\|^2+\lambda\rho_+(1-D(Aw-\gamma\mathbf{1})) , \tag{27}$$

Problem: Size	ADMM Iters	ADMM Inner	IP Iters	t_{ADMM} (s)	t_{IP} (s)	ObjDiff
Lasso: 1500×5000	15	—	18	2.0	58.3	0.0025
SVM: 32561×123						
$\text{cond}(A) = 7.7 \times 10^{10}$	653	—	77	41.2	23.9	0.17
H-Lasso: 1000×2000						
ADMM/ADMM						
$\text{cond}(A) = 5.8$	26	100	20	14.1	10.5	0.00006
$\text{cond}(A) = 1330$	27	100	24	40.0	13.0	0.0018
ADMM/L-BFGS						
$\text{cond}(A) = 5.8$	18	—	20	2.8	10.3	1.02
$\text{cond}(A) = 1330$	22	—	24	21.2	13.1	1.24
L1 Lasso: 500×2000						
ADMM/ADMM						
$\text{cond}(A) = 2.2$	104	100	29	57.4	5.9	0.06
$\text{cond}(A) = 1416$	112	100	29	81.4	5.6	0.21

Table 1: For each problem, we list iterations for IP, outer ADMM, cap for inner ADMM iterations (if applicable), total time for both algorithms and objective difference f(xADMM) - f(xIP). This difference is always positive, since in all experiments IP found a lower objective value, and its magnitude is an accuracy heuristic for ADMM, where lower difference means higher accuracy.

where ρ_+ is the hinge loss function, $w^T x = \gamma$ is the hyperplane being sought, $D \in \mathrm{R}^{m \times m}$ is a diagonal matrix with $\{\pm 1\}$ on the diagonals (in accordance to the classification of the training data), and $A \in \mathbb{R}^{m \times k}$ is the observation matrix, where each row gives the features corresponding to observation $i \in \{1, \ldots, m\}$. The ADMM details are similar to the Lasso example, so we omit them here. The interested reader can study the details in the file `linear_svm` available through `github/saravkin`.

The SVM example turned out to be very interesting. We downloaded the 9th Adult example from the SVM library at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`. The training set has 32561 examples, each with 123 features. When we formed the operator A for problem (27), we found it was very poorly conditioned, with condition number 7.7×10^{10}. It should not surprise the reader that after running for 653 iterations, ADMM is still appreciably far away—its objective value is higher, and in fact the relative norm distance to the (unique) true solution is 10%.

It is interesting to note that in this application, high optimization accuracy does not mean better classification accuracy on the test set—indeed, the (suboptimal) ADMM solution achieves a lower classification error on the test set (18%, vs. 18.75% error for IP). Nonetheless, this is not an advantage of one method over another—one can also stop the IP method early. The point here is that from the optimization perspective, SVM illustrates the advantages of Newton methods over methods with a linear rate.

5.3 Robust Lasso

For the examples in this section, we take $\rho(\cdot)$ to be a robust convex loss, either the 1-norm or the Huber function, and consider the robust Lasso problem

$$\min_x \rho(Ax-b)+\lambda\|x\|_1 \ .$$

First, we develop an ADMM approach that works for both losses, exploiting the simple nature of the regularizer. Then, we develop a second ADMM approach when $\rho(x)$ is the Huber function by exploiting partial smoothness of the objective.

Setting $z = Ax-b$, we obtain the augmented Lagrangian

$$\mathcal{L}(x,z,y) = \rho(z)+\lambda\|x\|_1+\eta y^T(z-Ax+b)+\frac{\rho}{2}\|-z+Ax-b\|_2^2 \ .$$

The ADMM updates for this formulation are

$$\begin{aligned}
x^{k+1} &= \operatorname*{argmin}_x \lambda\|x\|_1+\frac{\eta}{2}\|Ax-y^k-z^k\|_2^2,\\
z^{k+1} &= \operatorname*{argmin}_z \rho(z)+\frac{\eta}{2}\|z+y^k-Ax^{k+1}+b\|_2^2,\\
y^{k+1} &= y^k+(z^{k+1}-Ax^{k+1}+b) \ .
\end{aligned}$$

The z-update can be solved using thresholding, or modified thresholding, in $O(m)$ time when $\rho(\cdot)$ is the Huber loss or 1-norm. Unfortunately, the x-update now requires solving a LASSO problem. This can be done with ADMM (see previous section), but the nested ADMM structure does not perform as well as IP methods, even for well conditioned problems.

When $\rho(\cdot)$ is smooth, such as in the case of the Huber loss, the partial smoothness of the objective can be exploited by setting $x = z$, obtaining

$$\mathcal{L}(x,z,y) = \rho(Ax-b)+\lambda\|z\|_1+\eta y^T(zx)+\frac{\rho}{2}\|x-z\|_2^2 \ .$$

The ADMM updates are:

$$\begin{aligned}
x^{k+1} &= \operatorname*{argmin}_x \rho(Ax-b)+\frac{\eta}{2}\|x-z^k+y^k\|_2^2,\\
z^{k+1} &= \operatorname*{argmin}_z \lambda\|z\|_1+\frac{\eta}{2}\|z+(x^{k+1}+y^k)\|_2^2,\\
y^{k+1} &= y^k+(z^{k+1}-x^{k+1}) \ .
\end{aligned}$$

The problem required for the x-update is smooth, and can be solved by a fast quasi-Newton method, such as L-BFGS. L-BFGS is implemented using only matrix-vector products, and for well-conditioned problems, the ADMM/LBFGS approach has a speed advantage over IP methods. For ill-conditioned problems, L-BFGS has to work harder to achieve high accuracy, and inexact solves may destabilize the overall ADMM approach. IP methods are more consistent (see Table 1).

Just as in the Lasso problem, the IP implementation is dominated by the formation of A^TD^kA at every iteration with complexity $O(mn^2)$. However, a simple change of penalty makes the problem much harder for ADMM, especially when the operator A is ill-conditioned.

We hope that the toy problems, results, and code that we developed in order to write this section have given the reader a better intuition for IP methods. In the next section, we apply IP methods to Kalman smoothing, and show that these methods can be specifically designed to exploit the time series structure and preserve computational efficiency of classical Kalman smoothers.

6. Kalman Smoothing With PLQ Penalties

Consider now a dynamic scenario, where the system state x_k evolves according to the following stochastic discrete-time linear model

$$\begin{aligned} x_1 &= x_0 + w_1, \\ x_k &= G_k x_{k-1} + w_k, \qquad k = 2,3,\ldots,N \\ z_k &= H_k x_k + v_k, \qquad k = 1,2,\ldots,N \end{aligned} \tag{28}$$

where x_0 is known, z_k is the m-dimensional subvector of z containing the noisy output samples collected at instant k, G_k and H_k are known matrices. Further, we consider the general case where $\{w_k\}$ and $\{v_k\}$ are mutually independent zero-mean random variables which can come from any of the densities introduced in the previous section, with positive definite covariance matrices denoted by $\{Q_k\}$ and $\{R_k\}$, respectively.

In order to formulate the Kalman smoothing problem over the entire sequence $\{x_k\}$, define

$$\begin{aligned} x &= \text{vec}\{x_1,\cdots,x_N\}\,, & w &= \text{vec}\{w_1,\cdots,w_N\}\,, \\ v &= \text{vec}\{v_1,\cdots,v_N\}\,, & Q &= \text{diag}\{Q_1,\cdots,Q_N\}\,, \\ R &= \text{diag}\{R_1,\cdots,R_N\}\,, & H &= \text{diag}\{H_1,\cdots,H_N\}, \end{aligned}$$

and

$$G = \begin{bmatrix} \mathrm{I} & 0 & & \\ -G_2 & \mathrm{I} & \ddots & \\ & \ddots & \ddots & 0 \\ & & -G_N & \mathrm{I} \end{bmatrix}.$$

Then model (28) can be written in the form of (1)-(2), that is,

$$\begin{aligned} \mu &= Gx + w, \\ z &= Hx + v\,, \end{aligned} \tag{29}$$

where $x \in \mathbb{R}^{nN}$ is the entire state sequence of interest, w is corresponding process noise, z is the vector of all measurements, v is the measurement noise, and μ is a vector of size nN with the first n-block equal to x_0, the initial state estimate, and the other blocks set to 0. This is precisely the problem (1)-(2) that began our study. The problem (3) becomes the classical Kalman smoothing problem with quadratic penalties. In this case, the objective function can be written

$$\|Gx-\mu\|^2_{Q^{-1}} + \|Hx-z\|^2_{R^{-1}},$$

and the minimizer can be found by taking the gradient and setting it to zero:

$$(G^T Q^{-1} G + H^T R^{-1} H)x = r\,.$$

One can view this as a single step of Newton's method, which converges to the solution because the objective is quadratic. Note also that once the linear system above is formed, it takes only $O(n^3 N)$ operations to solve due to special block tridiagonal structure (for a generic system, it would take $O(n^3 N^3)$ time). In this section, we will show that IP methods can preserve this complexity for much more general penalties on the measurement and process residuals. We first make a brief remark related to the statistical interpretation of PLQ penalties.

Remark 16 *Suppose we decide to move to an outlier robust formulation, where the 1-norm or Huber penalties are used, but the measurement variance is known to be R. Using the statistical interpretation developed in section 3, the statistically correct objective function for the smoother is*

$$\frac{1}{2}\|Gx-\mu\|^2_{Q^{-1}} + \sqrt{2}\|R^{-1}(Hx-z)\|_1 \ .$$

Analogously, the statistically correct objective when measurement error is the Huber penalty with parameter κ *is*

$$\frac{1}{2}\|Gx-\mu\|^2_{Q^{-1}} + c_2\rho(R^{-1/2}(Hx-z)) \ ,$$

where

$$c_2 = \frac{4\exp\left[-\kappa^2/2\right]\frac{1+\kappa^2}{\kappa^3} + \sqrt{2\pi}[2\Phi(\kappa)-1]}{2\exp\left[-\kappa^2/2\right]\frac{1}{\kappa} + \sqrt{2\pi}[2\Phi(\kappa)-1]}.$$

The normalization constant comes from the results in Section 3.1, and ensures that the weighting between process and measurement terms is still consistent with the situation regardless of which shapes are used for the process and measurement penalties. This is one application of the statistical interpretation.

Next, we show that when the penalties used on the process residual $Gx-w$ and measurement residual $Hx-z$ arise from general PLQ densities, the general Kalman smoothing problem takes the form (20), studied in the previous section. The details are given in the following remark.

Remark 17 *Suppose that the noises w and v in the model* (29) *are PLQ densities with means* 0*, variances Q and R (see Def. 12). Then, for suitable* U_w, M_w, b_w, B_w *and* U_v, M_v, b_v, B_v *and corresponding* ρ_w *and* ρ_v *we have*

$$\begin{aligned}\mathbf{p}(w) &\propto \exp\left[-\rho\left(U_w, M_w, b_w, B_w; Q^{-1/2}w\right)\right], \\ \mathbf{p}(v) &\propto \exp\left[-\rho(U_v, M_v, b_v, B_v; R^{-1/2}v)\right]\end{aligned}$$

while the MAP estimator of x in the model (29) *is*

$$\underset{x\in\mathbb{R}^{nN}}{\operatorname{argmin}} \left\{ \begin{array}{l} \rho\left[U_w, M_w, b_w, B_w; Q^{-1/2}(Gx-\mu)\right] \\ +\rho\left[U_v, M_v, b_v, B_v; R^{-1/2}(Hx-z)\right] \end{array} \right\}. \tag{30}$$

If U_w and U_v are given as in (24), then the system (22) decomposes as

$$\begin{array}{rcll} 0 &=& A_w^{\mathrm{T}}u_w + s_w - a_w \ ; & 0 = A_v^{\mathrm{T}}u_v + s_v - a_v, \\ 0 &=& s_w^{\mathrm{T}}q_w \ ; & 0 = s_v^{\mathrm{T}}q_v, \\ 0 &=& \tilde{b}_w + B_wQ^{-1/2}Gd - M_wu_w - A_wq_w, & \\ 0 &=& \tilde{b}_v - B_vR^{-1/2}Hd - M_vu_v - A_vq_v, & \\ 0 &=& G^{\mathrm{T}}Q^{-\mathrm{T}/2}B_w^{\mathrm{T}}u_w - H^{\mathrm{T}}R^{-\mathrm{T}/2}B_v^{\mathrm{T}}u_v, & \\ 0 &\leq& s_w, s_v, q_w, q_v. & \end{array} \tag{31}$$

See the Appendix and (Aravkin, 2010) for details on deriving the KKT system. By further exploiting the decomposition shown in (28), we obtain the following theorem.

Theorem 18 (PLQ Kalman smoother theorem) *Suppose that all w_k and v_k in the Kalman smoothing model (28) come from PLQ densities that satisfy*

$$\text{null}(M_k^w) \cap \text{null}((A_k^w)^{\mathrm{T}}) = \{0\}\ , \text{null}(M_k^v) \cap \text{null}((A_k^v)^{\mathrm{T}}) = \{0\}\ ,\ \forall k\ ,$$

that is, their corresponding penalties are finite-valued. Suppose further that the corresponding set $\mathcal{F}_+$ from Theorem 14 is nonempty. Then (30) *can be solved using an IP method, with computational complexity $O[N(n^3+m^3+l)]$, where l is the largest column dimension of the matrices $\{A_k^{\mathrm{v}}\}$ and $\{A_k^w\}$.*

Note that the first part of this theorem, the solvability of the problem using IP methods, already follows from Theorem 14. The main contribution of the result in the dynamical system context is the computational complexity. The proof is presented in the Appendix and shows that IP methods for solving (30) preserve the key block tridiagonal structure of the standard smoother. If the number of IP iterations is fixed ($10-20$ are typically used in practice), general smoothing estimates can thus be computed in $O[N(n^3+m^3+l)]$ time. Notice also that the number of required operations scales linearly with l, which represents the complexity of the PLQ density encoding.

7. Numerical Example

In this section, we illustrate the modeling capabilities and computational efficiency of PLQ Kalman smoothers on simulated and real data.

7.1 Simulated Data

We use a simulated example to test the computational scheme described in the previous section. We consider the following function

$$f(t) = \exp[\sin(8t)]$$

taken from Dinuzzo et al. (2007). Our aim is to reconstruct f starting from 2000 noisy samples collected uniformly over the unit interval. The measurement noise v_k was generated using a mixture of two Gaussian densities, with $p = 0.1$ denoting the fraction from each Gaussian; that is,

$$v_k \sim (1-p)\mathbf{N}(0, 0.25) + p\mathbf{N}(0, 25),$$

Data are displayed as dots in Figure 2. Note that the purpose of the second component of the Gaussian mixture is to simulate outliers in the output data and that all the measurements exceeding vertical axis limits are plotted on upper and lower axis limits (4 and -2) to improve readability.

The initial condition $f(0) = 1$ is assumed to be known, while the difference of the unknown function from the initial condition (i.e., $f(\cdot) - 1$) is modeled as a Gaussian process given by an integrated Wiener process. This model captures the Bayesian interpretation of cubic smoothing splines (Wahba, 1990), and admits a two-dimensional state space representation where the first component of $x(t)$, which models $f(\cdot) - 1$, corresponds to the integral of the second state component, modelled as Brownian motion. To be more specific, letting $\Delta t = 1/2000$, the sampled version of the state space model (Jazwinski, 1970; Oksendal, 2005) is defined by

$$G_k = \begin{bmatrix} 1 & 0 \\ \Delta t & 1 \end{bmatrix}, \qquad k = 2, 3, \ldots, 2000$$
$$H_k = \begin{bmatrix} 0 & 1 \end{bmatrix}, \qquad k = 1, 2, \ldots, 2000$$

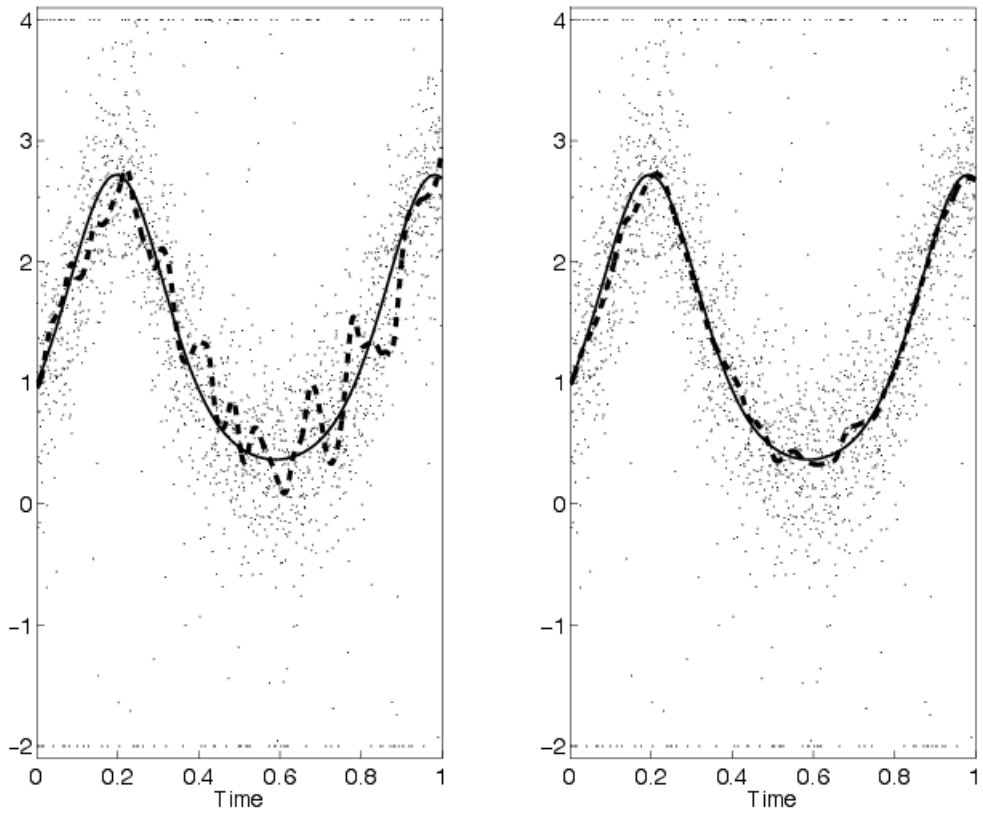


Figure 2: Simulation: measurements $(\cdot)$ with outliers plotted on axis limits (4 and -2), true function (continuous line), smoothed estimate using either the quadratic loss (dashed line, left panel) or the Vapnik's ε-insensitive loss (dashed line, right panel)

with the autocovariance of w_k given by

$$Q_k = \lambda^2 \begin{bmatrix} \Delta t & \frac{\Delta t^2}{2} \\ \frac{\Delta t^2}{2} & \frac{\Delta t^3}{3} \end{bmatrix}, \qquad k = 1, 2, \ldots, 2000\,,$$

where λ^2 is an unknown scale factor to be estimated from the data.

We compare the performance of two Kalman smoothers. The first (classical) estimator uses a quadratic loss function to describe the negative log of the measurement noise density and contains only λ^2 as unknown parameter. The second estimator is a Vapnik smoother relying on the ε-insensitive loss, and so depends on two unknown parameters: λ^2 and ε. In both cases, the unknown parameters are estimated by means of a cross validation strategy where the 2000 measurements are randomly split into a training and a validation set of 1300 and 700 data points, respectively. The Vapnik smoother was implemented by exploiting the efficient computational strategy described in the previous section; Aravkin et al. (2011b) provide specific implementation details. Efficiency is

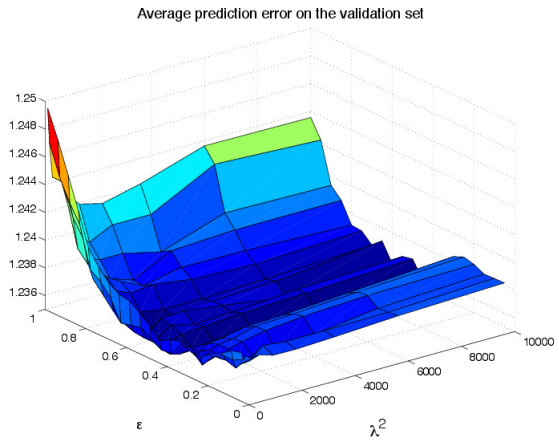


Figure 3: Estimation of the smoothing filter parameters using the Vapnik loss. Average prediction error on the validation data set as a function of the variance process λ^2 and ε.

particularly important here, because of the need for cross-validation. In this way, for each value of λ^2 and ε contained in a 10×20 grid on $[0.01, 10000] \times [0, 1]$, with λ^2 logarithmically spaced, the function estimate was rapidly obtained by the new smoother applied to the training set. Then, the relative average prediction error on the validation set was computed, see Figure 3. The parameters leading to the best prediction were $\lambda^2 = 2.15 \times 10^3$ and $\varepsilon = 0.45$, which give a sparse solution defined by fewer than 400 support vectors. The value of λ^2 for the classical Kalman smoother was then estimated following the same strategy described above. In contrast to the Vapnik penalty, the quadratic loss does not induce any sparsity, so that, in this case, the number of support vectors equals the size of the training set.

The left and right panels of Figure 2 display the function estimate obtained using the quadratic and the Vapnik losses, respectively. It is clear that the estimate obtained using the quadratic penalty is heavily affected by the outliers. In contrast, as expected, the estimate coming from the Vapnik based smoother performs well over the entire time period, and is virtually unaffected by the presence of large outliers.

7.2 Real Industrial Data

Let us now consider real industrial data coming from Syncrude Canada Ltd, also analyzed by Liu et al. (2004). Oil production data is typically a multivariate time series capturing variables such as flow rate, pressure, particle velocity, and other observables. Because the data is proprietary, the exact nature of the variables is not known. The data used by Liu et al. (2004) comprises two anonymized time series variables, called V14 and V36, that have been selected from the process data. Each time series consists of 936 measurements, collected at times $[1, 2, \ldots, 936]$ (see the top panels of Figure 4). Due to the nature of production data, we hypothesize that the temporal profile of the variables is smooth and that the observations contain outliers, as suggested by the fact that some observations differ markedly from their neighbors, especially in the case of V14.
Our aim is to compare the prediction performance of two smoothers that rely on ℓ_2 and ℓ_1 measure-

ment loss functions. For this purpose, we consider 100 Monte Carlo runs. During each run, data are randomly divided into three disjoint sets: training and a validation data sets, both of size 350, and a test set of size 236. We use the same state space model adopted in the previous subsection, with $\Delta t = 1$, and use a non-informative prior to model the initial condition of the system. The regularization parameter γ (equal to the inverse of λ^2 assuming that the noise variance is 1) is chosen using standard cross validation techniques. For each value of γ, logarithmically spaced between 0.1 and 1000 (30 point grid), the smoothers are trained on the training set, and the γ chosen corresponds to the smoother that achieves the best prediction on the validation set. After estimating γ, the variable's profile is reconstructed for the entire time series (at all times $[1,2,\ldots,936]$), using the measurements contained in the union of the training and the validation data sets. Then, the prediction capability of the smoothers is evaluated by computing the 236 relative percentage errors (ratio of residual and observation times 100) in the reconstruction of the test set.

In Figure 4 we display the boxplots of the overall 23600 relative errors stored after the 100 runs for V14 (bottom left panel) and V36 (bottom right panel). One can see that the ℓ_1-Kalman smoother outperforms the classical one, especially in case of V14. This is not surprising, since in this case prediction is more difficult due to the larger numbers of outliers in the time series. In particular, for V14, the average percentage errors are 1.4% and 2.4% while, for V36, they are 1% and 1.2% using ℓ_1 and ℓ_2, respectively.

8. Conclusions

We have presented a new theory for robust and sparse estimation using nonsmooth QS penalties. The QS class captures a variety of existing penalties, including all PLQ penalties, and we have shown how to construct natural generalizations based on general norm and cone geometries, and explored the structure of these functions using Euclidean projections.

Many penalties in the QS class can be interpreted as negative logs of true probability densities. Coercivity (characterized in Theorem 10) is the key necessary condition for this interpretation, as well as a fundamental prerequisite in sparse and robust estimation, since it precludes directions for which the sum of the loss and the regularizer are insensitive to large parameter changes. Thus, coercivity also ensures that the problem is well posed in the machine learning context, that is, the learning machine has enough control over model complexity.

It is straightforward to design new formulations in the QS framework. Starting with the requisite penalty *shape*, one can use results of Section 3 to obtain a standardized density, as well as the data structures required for the optimization problem in Section 4. The statistical interpretation for these methods allows specification of mean and variance parameters for the corresponding model.

In the second part of the paper, we presented a computational approach to solving estimation problems (4) using IP methods. We derived additional conditions that guarantee the successful implementation of IP methods to compute the estimator (4) when x and v come from PLQ densities, and characterized the convergence of IP methods for this class. The key condition for successful execution of IP iterations is for PLQ penalties to be finite valued, which implies non-degeneracy of the corresponding statistical distribution (the support cannot be contained in a lower-dimensional subspace). The statistical interpretation is thus strongly linked to the computational procedure.

We then applied both the statistical framework and the computational approach to the class of state estimation problems in discrete-time dynamic systems, extending the classical formulations to allow dynamics and measurement noise to come from any PLQ densities. We showed that clas-

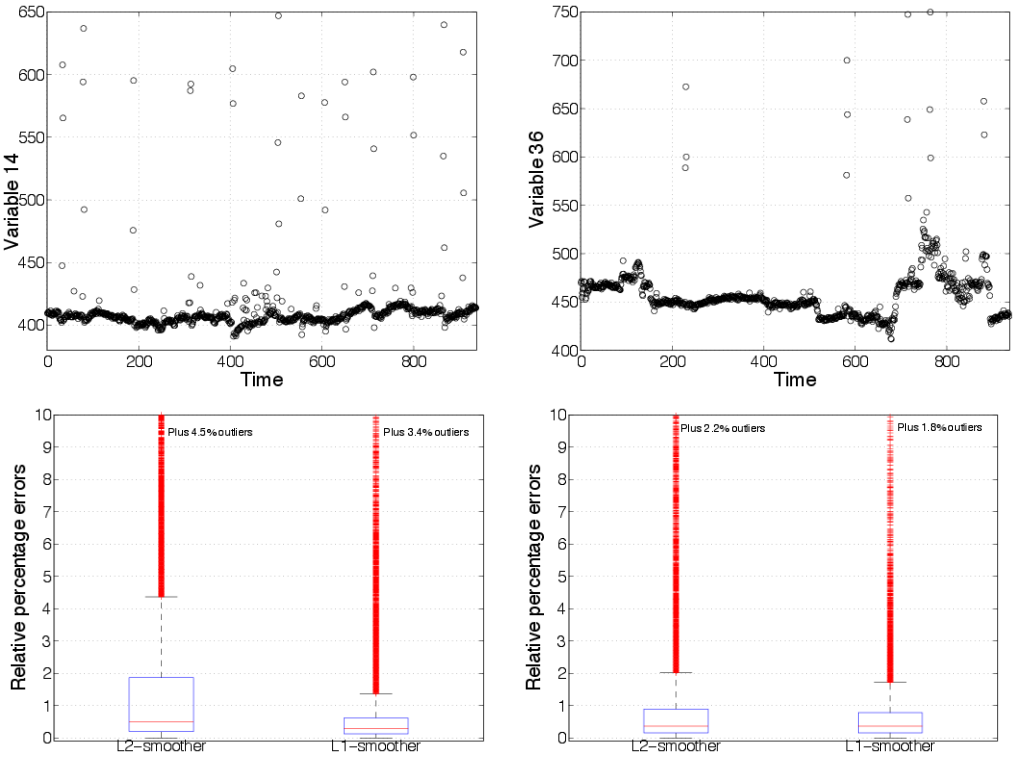


Figure 4: *Left panels*: data set for variable 14 (top) and relative percentage errors in the reconstruction of the test set obtained by Kalman smoothers based on the ℓ_2 and the ℓ_1 loss (bottom). *Right panels*: data set for variable 36 (top) and relative percentage errors in the reconstruction of the test set obtained by Kalman smoothers based on the ℓ_2 and the ℓ_1 loss (bottom).

sical computational efficiency results are preserved when the general IP approach is used for state estimation; specifically, the computational cost of PLQ Kalman smoothing scales linearly with the length of the time series, as in the quadratic case.

While we only considered convex formulations in this paper, the presented approach makes it possible to solve a much broader class of non-convex problems. In particular, if the functions Hx and Gx in (4) are replaced by nonlinear functions $g(x)$ and $h(x)$, the methods in this paper can be used to compute descent directions for the non-convex problem. For an example of this approach, see Aravkin et al. (2011a), which considers non-convex Kalman smoothing problems with nonlinear process and measurement models and solves by using the standard methodology of convex composite optimization (Burke, 1985). At each outer iteration the process and measurement models are linearized around the current iterate, and the descent direction is found by solving a particular subproblem of type (4) using IP methods.

In many contexts, it would be useful to estimate the parameters that define QS penalties; for example the κ in the Huber penalty or the ε in the Vapnik penalty. In the numerical examples presented in this paper, we have relied on cross-validation to accomplish this task. An alternative

method could be to compute the MAP points returned by our estimator for different filter parameters to gain information about the joint posterior of states and parameters. This strategy could help in designing a good proposal density for posterior simulation using, for example, particle smoothing filters (Ristic et al., 2004). We leave a detailed study of this approach to the QS modeling framework for future work.

Appendix A. Proofs

In this appendix, we present proofs for the new results given in the main body of the paper.

A.1 Proof of Theorem 3

Let $\rho(y) = \rho(U,M,I,0;y)$ so that $\rho(U,M,B,b;y) = \rho(b+By)$. Then $\mathrm{dom}\,(\rho(U,M,B,b;\cdot)) = B^{-1}(\mathrm{dom}\,(\rho)-b)$, hence the theorem follows if it can be shown that $\mathrm{bar}(U)+\mathrm{Ran}\,(M) \subset \mathrm{dom}\,(\rho) \subset [U^\infty \cap \mathrm{null}(M)]^\circ$ with equality when $\mathrm{bar}(U)+\mathrm{Ran}\,(M)$ is closed. Observe that if there exists $w \in U^\infty \cap \mathrm{null}(M)$ such that $\langle y, w\rangle > 0$, then trivially $\rho(y) = +\infty$ so $y \notin \mathrm{dom}\,(\rho)$. Consequently, $\mathrm{dom}\,(\rho) \subset [U^\infty \cap \mathrm{null}(M)]^\circ$. Next let $y \in \mathrm{bar}(U)+\mathrm{Ran}\,(M)$, then there is a $v \in \mathrm{bar}(U)$ and w such that $y = v+Mw$. Hence

$$\begin{aligned}
\sup_{u\in U}[\langle u, y\rangle - \tfrac{1}{2}\langle u, Mu\rangle] &= \sup_{u\in U}[\langle u, v+Mw\rangle - \tfrac{1}{2}\langle u, Mu\rangle] \\
&= \sup_{u\in U}[\langle u, v\rangle + \tfrac{1}{2}w^TMw - \tfrac{1}{2}(w-u)^TM(w-u)] \\
&\leq \delta^*(v\,|\,U) + \tfrac{1}{2}w^TMw \;<\; \infty\,.
\end{aligned}$$

Hence $\mathrm{bar}(U)+\mathrm{Ran}\,(M) \subset \mathrm{dom}\,(\rho)$.

If the set $\mathrm{bar}(U)+\mathrm{Ran}\,(M)$ is closed, then so is the set $\mathrm{bar}(U)$. Therefore, by (Rockafellar, 1970, Corollary 14.2.1), $(U^\infty)^\circ = \mathrm{bar}(U)$, and, by (Rockafellar, 1970, Corollary 16.4.2), $[U^\infty \cap \mathrm{null}(M)]^\circ = \mathrm{bar}(U)+\mathrm{Ran}\,(M)$, which proves the result.

In the polyhedral case, $\mathrm{bar}(U)$ is a polyhedral convex set, and the sum of such sets is also a polyhedral convex set (Rockafellar, 1970, Corollary 19.3.2).

A.2 Proof of Theorem 7

To see the first equation in (8) write $\rho(y) = \sup_u\left[\langle y, u\rangle - \left(\frac{1}{2}\|L^Tu\|_2^2 + \delta(u\,|\,U)\right)\right]$, and then apply the calculus of convex conjugate functions (Rockafellar, 1970, Section 16) to find that

$$\left(\tfrac{1}{2}\|L^T\cdot\|_2^2 + \delta(\cdot\,|\,U)\right)^*(y) = \inf_{s\in\mathbb{R}^k}\left[\tfrac{1}{2}\|s\|_2^2 + \delta^*(y-Ls\,|\,U)\right]\,.$$

The second equivalence in (8) follows from (Rockafellar, 1970, Theorem 14.5).

For the remainder, we assume that M is positive definite. In this case it is easily shown that $(MU)^\circ = M^{-1}U^\circ$. Hence, by Theorem 14.5 of Rockafellar (1970), $\gamma(\cdot\,|\,MU) = \delta^*\left(\cdot\,\middle|\,M^{-1}U^\circ\right)$. We use these facts freely throughout the proof.

The formula (9) follows by observing that

$$\tfrac{1}{2}\|s\|_2^2 + \delta^*(y-Ls\,|\,U) = \tfrac{1}{2}\|L^{-T}s\|_M^2 + \delta^*\left(M^{-1}y - L^{-T}s\,|\,MU\right)$$

and then making the substitution $v = L^{-T}s$. To see (10), note that the optimality conditions for (9) are $Ms \in \partial\delta^*\left(M^{-1}y - s \,|\, MU\right)$, or equivalently, $M^{-1}y - s \in N\left(Ms \,|\, MU\right)$, that is, $s \in U$ and

$$\left\langle M^{-1}y - s, u - s\right\rangle_M = \left\langle M^{-1}y - s, M(u-s)\right\rangle \leq 0 \;\forall\, u \in U,$$

which, by (7), tells us that $s = P_M\left(M^{-1}y \,|\, U\right)$. Plugging this into (9) gives (10).

Using the substitution $v = Ls$, the argument showing (11) and (12) differs only slightly from that for (9) and (10) and so is omitted.

The formula (13) follows by completing the square in the M-norm in the definition (5):

$$\begin{aligned}
\langle y, u\rangle - \tfrac{1}{2}\langle u, Mu\rangle &= \left\langle M^{-1}y, u\right\rangle_M - \tfrac{1}{2}\langle u, u\rangle_M \\
&= \tfrac{1}{2}y^T M^{-1}y - \tfrac{1}{2}[\left\langle M^{-1}y, M^{-1}y\right\rangle_M - 2\left\langle M^{-1}y, u\right\rangle_M + \langle u, u\rangle_M] \\
&= \tfrac{1}{2}y^T M^{-1}y - \tfrac{1}{2}\|M^{-1}y - u\|_M^2 \,.
\end{aligned}$$

The result as well as (14) now follow from Theorem 6. Both (15) and (16) follow similary by completing the square in the M^{-1}-norm.

A.3 Proof of Theorem 9

First we will show that if ρ is convex coercive, then for any $\bar{x} \in \operatorname{argmin} f \neq \emptyset$, there exist constants R and $K > 0$ such that

$$\rho(x) \geq \rho(\bar{x}) + K\|x - \bar{x}\| \quad \forall\, x \notin R\mathbb{B}\,. \tag{32}$$

Without loss of generality, we can assume that $0 = \rho(0) = \inf\rho$. Otherwise, replace $\rho(x)$ by $\hat{\rho}(x) = \rho(x + \bar{x}) - \rho(\bar{x})$, where $\bar{x}$ is any global minimizer of ρ.

Let $\alpha > 0$. Since ρ is coercive, there exists R such that $\operatorname{lev}_\rho(\alpha) \subset R\mathbb{B}$. We will show that $\frac{\alpha}{R}\|x\| \leq \rho(x)$ for all $x \notin R\mathbb{B}$.

Indeed, for all $x \neq 0$, we have $\rho(\frac{R}{\|x\|}x) \geq \alpha$. Therefore, if $x \notin R\mathbb{B}$, then $0 < \frac{R}{\|x\|} < 1$, and we have

$$\frac{\alpha}{R}\|x\| \leq \frac{\|x\|}{R}\rho\left(\frac{R}{\|x\|}x\right) \leq \frac{\|x\|}{R}\frac{R}{\|x\|}\rho(x) = \rho(x).$$

Then by (32),

$$\begin{aligned}
\int \exp(-\rho(x))dx &= \int_{\bar{x}+R\mathbb{B}} \exp(-\rho(x))dx + \int_{\|x-\bar{x}\|>R} \exp(-\rho(x))dx \\
&\leq C_1 + C_2 \int_{\|x-\bar{x}\|>R} \exp(-K\|x-\bar{x}\|)dx < \infty\,.
\end{aligned}$$

A.4 Proof of Theorem 10

First observe that $B^{-1}[\operatorname{cone}(U)]^\circ = [B^{\mathrm{T}}\operatorname{cone}(U)]^\circ$ by Corollary 16.3.2 of Rockafellar (1970).

Suppose that $\hat{y} \in B^{-1}[\operatorname{cone}(U)]^\circ$, and $\hat{y} \neq 0$. Then $B\hat{y} \in \operatorname{cone}(U)$, and $B\hat{y} \neq 0$ since B is injective, and we have

$$\begin{aligned}
\rho(t\hat{y}) &= \sup_{u\in U}\langle b + tB\hat{y}, u\rangle - \tfrac{1}{2}u^{\mathrm{T}}Mu \\
&= \sup_{u\in U}\langle b, u\rangle - \tfrac{1}{2}u^{\mathrm{T}}Mu + t\langle B\hat{y}, u\rangle \\
&\leq \sup_{u\in U}\langle b, u\rangle - \tfrac{1}{2}u^{\mathrm{T}}Mu \\
&\leq \rho(U, M, 0, I; b),
\end{aligned}$$

so $\rho(t\hat{y})$ stays bounded even as $t \to \infty$, and so ρ cannot be coercive.

Conversely, suppose that ρ is not coercive. Then we can find a sequence $\{y_k\}$ with $\|y_k\| > k$ and a constant P so that $\rho(y_k) \leq P$ for all $k > 0$. Without loss of generality, we may assume that $\frac{y_k}{\|y_k\|} \to \bar{y}$.

Then by definition of ρ, we have for all $u \in U$

$$\begin{array}{l} \langle b + By_k, u\rangle - \frac{1}{2}u^{\mathrm{T}}Mu \leq P, \\ \langle b + By_k, u\rangle \leq P + \frac{1}{2}u^{\mathrm{T}}Mu, \\ \langle \frac{b+By_k}{\|y_k\|}, u\rangle \leq \frac{P}{\|y_k\|} + \frac{1}{2\|y_k\|}u^{\mathrm{T}}Mu. \end{array}$$

Note that $\bar{y} \neq 0$, so $B\bar{y} \neq 0$. When we take the limit as $k \to \infty$, we get $\langle B\bar{y}, u\rangle \leq 0$. From this inequality we see that $B\bar{y} \in [\mathrm{cone}(U)]^\circ$, and so $\bar{y} \in B^{-1}[\mathrm{cone}(U)]^\circ$.

A.5 Proof of Theorem 14

Proof (i) Using standard elementary row operations, reduce the matrix

$$F_\gamma^{(1)} := \begin{bmatrix} I & 0 & A^{\mathrm{T}} & 0 \\ D(q) & D(s) & 0 & 0 \\ 0 & -A & -M & B \\ 0 & 0 & B^{\mathrm{T}} & 0 \end{bmatrix}$$

to

$$\begin{bmatrix} I & 0 & A^{\mathrm{T}} & 0 \\ 0 & D(s) & -D(q)A^{\mathrm{T}} & 0 \\ 0 & 0 & -T & B \\ 0 & 0 & B^{\mathrm{T}} & 0 \end{bmatrix},$$

where $T = M + AD(q)D(s)^{-1}A^{\mathrm{T}}$. The matrix T is invertible since $\mathrm{null}(M) \cap \mathrm{null}(C^{\mathrm{T}}) = \{0\}$. Hence, we can further reduce this matrix to the block upper triangular form

$$\begin{bmatrix} I & 0 & A^{\mathrm{T}} & 0 \\ 0 & D(s) & -D(q)C^{\mathrm{T}} & 0 \\ 0 & 0 & -T & B \\ 0 & 0 & 0 & -B^{\mathrm{T}}T^{-1}B \end{bmatrix}.$$

Since B is injective, the matrix $B^{\mathrm{T}}T^{-1}B$ is also invertible. Hence this final block upper triangular is invertible proving Part (i).

(ii) Let $(s,q) \in \widehat{\mathcal{F}}_+$ and choose (u_i, y_i) so that $(s,q,u_i,y_i) \in \mathcal{F}_+$ for $i = 1,2$. Set $u := u_1 - u_2$ and $y := y_1 - y_2$. Then, by definition,

$$0 = A^{\mathrm{T}}u,\ 0 = By - Mu, \text{ and } 0 = B^{\mathrm{T}}u\ . \tag{33}$$

Multiplying the second of these equations on the left by u and using the third as well as the positive semi-definiteness of M, we find that $Mu = 0$. Hence, $u \in \mathrm{null}(M) \cap \mathrm{null}(A^{\mathrm{T}}) = \{0\}$, and so $By = 0$. But then $y = 0$ as B is injective.

(iii) Let $(\hat{s},\hat{q},\hat{u},\hat{y}) \in \mathcal{F}_+$ and $(s,q,u,y) \in \mathcal{F}_+(\tau)$. Then, by (22),

$$\begin{aligned}(s-\hat{s})^{\mathrm{T}}(q-\hat{q}) &= [(a-A^{\mathrm{T}}u)-(a-A^{\mathrm{T}}\hat{u})]^{\mathrm{T}}(q-\hat{q}) \\ &= (\hat{u}-u)^{\mathrm{T}}(Aq-A\hat{q}) \\ &= (\hat{u}-u)^{\mathrm{T}}[(b+By-Mu)-(b+B\hat{b}-M\hat{u})] \\ &= (\hat{u}-u)^{\mathrm{T}}M(\hat{u}-u) \\ &\geq 0.\end{aligned}$$

Hence,

$$\tau+\hat{s}^{\mathrm{T}}\hat{q} \geq s^{\mathrm{T}}y+\hat{s}^{\mathrm{T}}\hat{q} \geq s^{\mathrm{T}}\hat{y}+y^{\mathrm{T}}\hat{s} \geq \xi\,\|(s,q)\|_1\,,$$

where $\xi = \min\{\hat{s}_i,\ \hat{q}_i \mid i=1,\ldots,\ell\} > 0$. Therefore, the set

$$\widehat{\mathcal{F}}_+(\tau) = \{(s,q) \mid (s,q,u,y) \in \mathcal{F}_+(\tau)\}$$

is bounded. Now suppose the set $\mathcal{F}_+(\tau)$ is not bounded. Then there exists a sequence $\{(s_\nu,q_\nu,u_\nu,y_\nu)\} \subset \mathcal{F}_+(\tau)$ such that $\|(s_\nu,q_\nu,u_\nu,y_\nu)\| \uparrow +\infty$. Since $\widehat{\mathcal{F}}_+(\tau)$ is bounded, we can assume that $\|(u_\nu,y_\nu)\| \uparrow +\infty$ while $\|(s_\nu,q_\nu)\|$ remains bounded. With no loss in generality, we may assume that there exits $(u,y) \neq (0,0)$ such that $(u_\nu,y_\nu)/\|(u_\nu,y_\nu)\| \to (u,y)$. By dividing (22) by $\|(u_\nu,y_\nu)\|$ and taking the limit, we find that (33) holds. But then, as in (33), $(u,y) = (0,0)$. This contradiction yields the result.

(iv) We first show existence. This follows from a standard continuation argument. Let $(\hat{s},\hat{q},\hat{u},\hat{y}) \in \mathcal{F}_+$ and $v \in \mathbb{R}^\ell_{++}$. Define

$$F(s,q,u,y,t) = \begin{bmatrix} s+A^{\mathrm{T}}u-a \\ D(q)D(s)\mathbf{1}-[(1-t)\hat{v}+tv] \\ By-Mu-Aq \\ B^{\mathrm{T}}u+b \end{bmatrix},$$

where $\hat{g} := (\hat{s}_1\hat{y}_1,\ldots,\hat{s}_\ell\hat{y}_\ell)^{\mathrm{T}}$. Note that

$$F(\hat{s},\hat{q},\hat{u},\hat{y},0) = 0 \text{ and, by Part (i), } \nabla_{(s,q,u,y)}F(\hat{s},\hat{q},\hat{u},\hat{y},0)^{-1} \text{ exists.}$$

The Implicit Function Theorem implies that there is a $\tilde{t} > 0$ and a differentiable mapping $t \mapsto (s(t),q(t),u(t),y(t))$ on $[0,\tilde{t})$ such that

$$F[s(t),q(t),u(t),y(t),t] = 0 \text{ on } [0,\tilde{t}).$$

Let $\bar{t} > 0$ be the largest such $\tilde{t}$ on $[0,1]$. Since

$$\{[s(t),q(t),u(t),y(t)] \mid t \in [0,\bar{t})\} \subset \mathcal{F}_+(\bar{\tau}),$$

where $\bar{\tau} = \max\{\mathbf{1}^{\mathrm{T}}\hat{g},\mathbf{1}^{\mathrm{T}}g\}$, Part (iii) implies that there is a sequence $t_i \to \bar{t}$ and a point $(\bar{s},\bar{q},\bar{u},\bar{y})$ such that $[s(t_i),q(t_i),u(t_i),y(t_i)] \to (\bar{s},\bar{q},\bar{u},\bar{y})$. By continuity $F(\bar{s},\bar{q},\bar{u},\bar{y},\bar{t}) = 0$. If $\bar{t} = 1$, we are done; otherwise, apply the Implicit Function Theorem again at $(\bar{s},\bar{q},\bar{u},\bar{y},\bar{t})$ to obtain a contradiction to the maximality of $\bar{t}$.

We now show uniqueness. By Part (ii), we need only establish the uniqueness of (s,q). Let $(s^\nu,q^\nu) \in \widehat{\mathcal{F}}_+$ be such that $g = (s_{j(1)}q_{j(1)},s_{j(2)}q_{j(2)},\ldots,s_{j(\ell)}q_{j(\ell)})^{\mathrm{T}}$, where $s_{j(i)}$ denotes the ith element of s_j, and $j = 1,2$. As in Part (iii), we have $(s_1-s_2)^{\mathrm{T}}(q_1-q_2) = (u_1-u_2)^{\mathrm{T}}M((u_1-u_2) \geq 0$,

and, for each $i = 1,\ldots,\ell$, $s_{1(i)}q_{1(i)} = s_{2(i)}q_{2(i)} = g_i > 0$. If $(s_1,q_1) \neq (s_2,q_2)$, then, for some $i \in \{1,\ldots,\ell\}$, $(s_{1(i)} - s_{2(i)})(q_{1(i)} - q_{2(i)}) \geq 0$ and either $s_{1(i)} \neq s_{2(i)}$ or $q_{1(i)} \neq q_{2(i)}$. If $s_{1(i)} > s_{2(i)}$, then $q_{1(i)} \geq q_{2(i)} > 0$ so that $g_i = s_{1(i)}q_{1(i)} > s_{2(i)}q_{2(i)} = g_i$, a contradiction. So with out loss in generality (by exchanging (s_1,q_1) with (s_2,q_2) if necessary), we must have $q_{1(i)} > q_{2(i)}$. But then $s_{1(i)} \geq s_{2(i)} > 0$, so that again $g_i = s_{1(i)}q_{1(i)} > s_{2(i)}q_{2(i)} = g_i$, and again a contradiction. Therefore, (s,q) is unique.
(v) Apply Part (iv) to get a point on the central path and then use the continuation argument to trace out the central path. The differentiability follows from the implicit function theorem.
(vi) Part (iii) allows us to apply a standard compactness argument to get the existence of cluster points and the continuity of $F_\gamma(s,q,u,y)$ in all of its arguments including γ implies that all of these cluster points solve (22). ■

A.6 Details for Remark 17

The Lagrangian for (30) for feasible (x,u_w,u_v) is

$$L(x,u_w,u_v) = \left\langle \begin{bmatrix} \tilde{b}_w \\ \tilde{b}_v \end{bmatrix}, \begin{bmatrix} u_w \\ u_v \end{bmatrix} \right\rangle - \frac{1}{2}\begin{bmatrix} u_w \\ u_v \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} M_w & 0 \\ 0 & M_v \end{bmatrix} \begin{bmatrix} u_w \\ u_v \end{bmatrix} - \left\langle \begin{bmatrix} u_w \\ u_v \end{bmatrix}, \begin{bmatrix} -B_wQ^{-1/2}G \\ B_vR^{-1/2}H \end{bmatrix} x \right\rangle$$

where $\tilde{b}_w = b_w - B_wQ^{-1/2}\tilde{x}_0$ and $\tilde{b}_v = b_v - B_vR^{-1/2}z$. The associated optimality conditions for feasible (x,u_w,u_v) are given by

$$\begin{array}{l} G^{\mathrm{T}}Q^{-\mathrm{T}/2}B_w^{\mathrm{T}}\bar{u}_w - H^{\mathrm{T}}R^{-\mathrm{T}/2}B_v^{\mathrm{T}}\bar{u}_v = 0, \\ \tilde{b}_w - M_w\bar{u}_w + B_wQ^{-1/2}G\bar{x} \in N_{U_w}(\bar{u}_w), \\ \tilde{b}_v - M_v\bar{u}_v - B_vR^{-1/2}H\bar{x} \in N_{U_v}(\bar{u}_v)\ , \end{array} \tag{34}$$

where $N_C(r)$ denotes the normal cone to the set C at the point r (Rockafellar, 1970).

Since U_w and U_v are polyhedral, we can derive explicit representations of the normal cones $N_{U_w}(\bar{u}_w)$ and $N_{U_v}(\bar{u}_v)$. For a polyhedral set $U \subset \mathbb{R}^m$ and any point $\bar{u} \in U$, the normal cone $N_U(\bar{u})$ is polyhedral. Indeed, relative to any representation

$$U = \{u | A^{\mathrm{T}}u \leq a\}$$

and the active index set $I(\bar{u}) := \{i | \langle A_i, \bar{u}\rangle = a_i\}$, where A_i denotes the ith column of A, we have

$$N_U(\bar{u}) = \left\{ q_1A_1 + \cdots + q_mA_m \;\middle|\; q_i \geq 0 \text{ for } i \in I(\bar{u}), \quad q_i = 0 \text{ for } i \notin I(\bar{u}) \right\}. \tag{35}$$

Using (35), Then we may rewrite the optimality conditions (34) more explicitly as

$$\begin{array}{l} G^{\mathrm{T}}Q^{-\mathrm{T}/2}B_w^{\mathrm{T}}\bar{u}_w - H^{\mathrm{T}}R^{-\mathrm{T}/2}B_v^{\mathrm{T}}\bar{u}_v = 0, \\ \tilde{b}_w - M_w\bar{u}_w + B_wQ^{-1/2}G\bar{d} = A_wq_w, \\ \tilde{b}_v - M_v\bar{u}_v - B_vR^{-1/2}H\bar{d} = A_vq_v, \\ \{q_v \geq 0 | q_{v(i)} = 0 \text{ for } i \notin I(\bar{u}_v)\}, \\ \{q^w \geq 0 | q_{w(i)} = 0 \text{ for } i \notin I(\bar{u}_w)\}\ . \end{array}$$

where $q_{v(i)}$ and $q_{w(i)}$ denote the ith elements of q_v and q_w. Define slack variables $s_w \geq 0$ and $s_v \geq 0$ as follows:

$$\begin{aligned} s_w &= a_w - A_w^{\mathrm{T}} u_w, \\ s_v &= a_v - A_v^{\mathrm{T}} u_v. \end{aligned}$$

Note that we know the entries of $q_{w(i)}$ and $q_{v(i)}$ are zero if and only if the corresponding slack variables $s_{v(i)}$ and $s_{w(i)}$ are nonzero, respectively. Then we have $q_w^{\mathrm{T}} s_w = q_v^{\mathrm{T}} s_v = 0$. These equations are known as the complementarity conditions. Together, all of these equations give system (31).

A.7 Proof of Theorem 18

IP methods apply a damped Newton iteration to find the solution of the relaxed KKT system $F_\gamma = 0$, where

$$F_\gamma \begin{pmatrix} s_w \\ s_v \\ q_w \\ q_v \\ u_w \\ u_v \\ x \end{pmatrix} = \begin{bmatrix} A_w^{\mathrm{T}} u_w + s_w - a_w \\ A_v^{\mathrm{T}} u_v + s_v - a_v \\ D(q_w) D(s_w) \mathbf{1} - \gamma \mathbf{1} \\ D(q_v) D(s_v) \mathbf{1} - \gamma \mathbf{1} \\ \tilde{b}_w + B_w Q^{-1/2} G d - M_w u_w - A_w q_w \\ \tilde{b}_v - B_v R^{-1/2} H d - M_v u_v - A_v q_v \\ G^{\mathrm{T}} Q^{-\mathrm{T}/2} B_w^{\mathrm{T}} u_w - H^{\mathrm{T}} R^{-\mathrm{T}/2} B_v^{\mathrm{T}} \bar{u}_v \end{bmatrix}.$$

This entails solving the system

$$F_\gamma^{(1)} \begin{pmatrix} s_w \\ s_v \\ q_w \\ q_v \\ u_w \\ u_v \\ x \end{pmatrix} \begin{bmatrix} \Delta s_w \\ \Delta s_v \\ \Delta q_w \\ \Delta q_v \\ \Delta u_w \\ \Delta u_v \\ \Delta x \end{bmatrix} = -F_\gamma \begin{pmatrix} s_w \\ s_v \\ q_w \\ q_v \\ u_w \\ u_v \\ x \end{pmatrix}, \tag{36}$$

where the derivative matrix $F_\gamma^{(1)}$ is given by

$$\begin{bmatrix} I & 0 & 0 & 0 & (A_w)^{\mathrm{T}} & 0 & 0 \\ 0 & I & 0 & 0 & 0 & (A_v)^{\mathrm{T}} & 0 \\ D(q_w) & 0 & D(s_w) & 0 & 0 & 0 & 0 \\ 0 & D(q_v) & 0 & D(s_v) & 0 & 0 & 0 \\ 0 & 0 & -A_w & 0 & -M_w & 0 & B_w Q^{-1/2} G \\ 0 & 0 & 0 & -A_v & 0 & -M_v & -B_v R^{-1/2} H \\ 0 & 0 & 0 & 0 & G^{\mathrm{T}} Q^{-\mathrm{T}/2} B_w^{\mathrm{T}} & -H^{\mathrm{T}} R^{-\mathrm{T}/2} B_v^{\mathrm{T}} & 0 \end{bmatrix}. \tag{37}$$

We now show the row operations necessary to reduce the matrix $F_\gamma^{(1)}$ in (37) to upper block triangular form. After each operation, we show only the row that was modified.

$$\begin{array}{l}
\text{row}_3 \leftarrow \text{row}_3 - D(q_w)\,\text{row}_1 \\
\begin{bmatrix} 0 & 0 & D(s_w) & 0 & -D(q_w)A_w^{\mathrm{T}} & 0 & 0 \end{bmatrix} \\
\text{row}_4 \leftarrow \text{row}_4 - D(q_v)\,\text{row}_2 \\
\begin{bmatrix} 0 & 0 & 0 & D(s_v) & 0 & -D(q_v)A_v^{\mathrm{T}} & 0 \end{bmatrix} \\
\text{row}_5 \leftarrow \text{row}_5 + A_w D(s_w)^{-1}\,\text{row}_3 \\
\begin{bmatrix} 0 & 0 & 0 & 0 & -T_w & 0 & B_w Q^{-1/2} G \end{bmatrix} \\
\text{row}_6 \leftarrow \text{row}_6 + A_v D(s_v)^{-1}\,\text{row}_4 \\
\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -T_v & -B_v R^{-1/2} H \end{bmatrix} .
\end{array}$$

In the above expressions,

$$\begin{aligned}
T_w &:= M_w + A_w D(s_w)^{-1} D(q_w) A_w^{\mathrm{T}}, \\
T_v &:= M_v + A_v D(s_v)^{-1} D(q_v) A_v^{\mathrm{T}} ,
\end{aligned} \tag{38}$$

where $D(s_w)^{-1}D(q_w)$ and $D(s_v)^{-1}D(q_v)$ are always full-rank diagonal matrices, since the vectors s_w, q_w, s_v, q_v. Matrices T_w and T_v are invertible as long as the PLQ densities for w and v satisfy (25).

Remark 19 (block diagonal structure of T in i.d. case) *Suppose that y is a random vector, $y = \text{vec}(\{y_k\})$, where each y_i is itself a random vector in $\mathbb{R}^{m(i)}$, from some PLQ density $\mathbf{p}(y_i) \propto \exp[-c_2\mathsf{p}(U_i, M_i, 0, I; \cdot)]$, and all y_i are independent. Let $U_i = \{u : A_i^{\mathrm{T}} u \le a_i\}$. Then the matrix T_p is given by $T_\mathsf{p} = M + ADA^{\mathrm{T}}$ where $M = \text{diag}[M_1, \cdots, M_N]$, $A = \text{diag}[A_1, \cdots, A_N]$, $D = \text{diag}[D_1, \cdots, D_N]$, and $\{D_i\}$ are diagonal with positive entries. Moreover, T_p is block diagonal, with ith diagonal block given by $M_i + A_i D_i A_i^{\mathrm{T}}$.*

From Remark 19, the matrices T_w and T_v in (38) are block diagonal provided that $\{w_k\}$ and $\{v_k\}$ are independent vectors from any PLQ densities.

We now finish the reduction of $F_\gamma^{(1)}$ to upper block triangular form:

$$\text{row}_7 \leftarrow \text{row}_7 + \left(G^{\mathrm{T}} Q^{-\mathrm{T}/2} B_w^{\mathrm{T}} T_w^{-1}\right)\text{row}_5 - \left(H^{\mathrm{T}} R^{-\mathrm{T}/2} B_v^{\mathrm{T}} T_v^{-1}\right)\text{row}_6$$

$$\begin{bmatrix}
I & 0 & 0 & 0 & (A_w)^{\mathrm{T}} & 0 & 0 \\
0 & I & 0 & 0 & 0 & (A_v)^{\mathrm{T}} & 0 \\
0 & 0 & S_w & 0 & -Q_w(A_w)^{\mathrm{T}} & 0 & 0 \\
0 & 0 & 0 & S_v & 0 & -Q_v(A_v)^{\mathrm{T}} & 0 \\
0 & 0 & 0 & 0 & -T_w & 0 & B_w Q^{-1/2} G \\
0 & 0 & 0 & 0 & 0 & -T_v & -B_v R^{-1/2} H \\
0 & 0 & 0 & 0 & 0 & 0 & \Omega
\end{bmatrix}$$

where

$$\Omega = \Omega_G + \Omega_H = G^{\mathrm{T}} Q^{-\mathrm{T}/2} B_w^{\mathrm{T}} T_w^{-1} B_w Q^{-1/2} G + H^{\mathrm{T}} R^{-\mathrm{T}/2} B_v^{\mathrm{T}} T_v^{-1} B_v R^{-1/2} H.$$

Note that Ω is symmetric positive definite. Note also that Ω is block tridiagonal, since

1. Ω_H is block diagonal.

2. $Q^{-\mathrm{T}/2}B_w^{\mathrm{T}}T_w^{-1}B_wQ^{-1/2}$ is block diagonal, and G is block bidiagonal, hence Ω_G is block tridiagonal.

Solving system (36) requires inverting the block diagonal matrices T_v and T_w at each iteration of the damped Newton's method, as well as solving an equation of the form $\Omega\Delta x = \rho$. The matrices T_v and T_w are block diagonal, with sizes Nn and Nm, assuming m measurements at each time point. Given that they are invertible (see (25)), these inversions take $O(Nn^3)$ and $O(Nm^3)$ time. Since Ω is block tridiagonal, symmetric, and positive definite, $\Omega\Delta x = \rho$ can be solved in $O(Nn^3)$ time using the block tridiagonal algorithm in Bell (2000). The remaining four back solves required to solve (36) can each be done in $O(Nl)$ time, where we assume that $A_{v(k)} \in \mathbb{R}^{n\times l}$ and $A_{w(k)} \in \mathbb{R}^{m\times l}$ at each time point k.

References

B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, N.J., USA, 1979.

A. Y. Aravkin, J. V. Burke, and M. P. Friedlander. Variational properties of value functions. *To Appear in Siam Journal of Optimization*, 2013.

A.Y. Aravkin. *Robust Methods with Applications to Kalman Smoothing and Bundle Adjustment*. PhD thesis, University of Washington, Seattle, WA, June 2010.

A.Y. Aravkin, B.M. Bell, J.V. Burke, and G. Pillonetto. An ℓ_1-laplace robust kalman smoother. *Automatic Control, IEEE Transactions on*, 56(12):2898–2911, dec. 2011a. ISSN 0018-9286. doi: 10.1109/TAC.2011.2141430.

A.Y. Aravkin, B.M. Bell, J.V. Burke, and G. Pillonetto. Learning using state space kernel machines. In *Proc. IFAC World Congress 2011*, Milan, Italy, 2011b.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

B.M. Bell. The marginal likelihood for parameters in a discrete Gauss-Markov process. *IEEE Transactions on Signal Processing*, 48(3):626–636, August 2000.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1): 1–122, January 2011. ISSN 1935-8237. doi: 10.1561/2200000016. URL `http://dx.doi.org/10.1561/2200000016`.

R. Brockett. *Finite Dimensional Linear Systems*. John Wiley and Sons, Inc., 1970.

J. V. Burke. An exact penalization viewpoint of constrained optimization. Technical report, Argonne National Laboratory, ANL/MCS-TM-95, 1987.

J.V. Burke. Descent methods for composite nondifferentiable optimization problems. *Mathematical Programming*, 33:260–279, 1985.

W. Chu, S. S. Keerthi, and O. C. Jin. A unified loss function in bayesian framework for support vector regression. In *In Proceeding of the 18th International Conference on Machine Learning*, pages 51–58, 2001.

F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2001.

F. Dinuzzo. Analysis of fixed-point and coordinate descent algorithms for regularized kernel methods. *IEEE Transactions on Neural Networks*, 22(10):1576 –1587, 2011.

F. Dinuzzo, M. Neve, G. De Nicolao, and U. P. Gianazza. On the representer theorem and equivalent degrees of freedom of SVR. *Journal of Machine Learning Research*, 8:2467–2495, 2007.

D. Donoho. Compressed sensing. *IEEE Trans. on Information Theory*, 52(4):1289–1306, 2006.

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

T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–150, 2000.

S. Farahmand, G.B. Giannakis, and D. Angelosante. Doubly robust smoothing of dynamical processes via outlier sparsity constraints. *IEEE Transactions on Signal Processing*, 59:4529–4543, 2011.

M.C. Ferris and T.S. Munson. Interior-point methods for massive support vector machines. *SIAM Journal on Optimization*, 13(3):783 – 804, 2003.

S. Fine and K. Scheinberg. Efficient svm training using low-rank kernel representations. *J. Mach. Learn. Res.*, 2:243 –264, 2001.

J. Gao. Robust l1 principal component analysis and its Bayesian variational inference. *Neural Computation*, 20(2):555–572, February 2008.

A. Gelb. *Applied Optimal Estimation*. The M.I.T. Press, Cambridge, MA, 1974.

O. Güler and R. Hauser. Self-scaled barrier functions on symmetric cones and their classification. *Foundations of Computational Mathematics*, 2:121–143, 2002.

T. J. Hastie and R. J. Tibshirani. Generalized additive models. In *Monographs on Statistics and Applied Probability*, volume 43. Chapman and Hall, London, UK, 1990.

T. J. Hastie, R. J. Tibshirani, and J. Friedman. *The Elements of Statistical Learning. Data Mining, Inference and Prediction*. Springer, Canada, 2001.

P.J. Huber. *Robust Statistics*. Wiley, 1981.

A. Jazwinski. *Stochastic Processes and Filtering Theory*. Dover Publications, Inc, 1970.

T. Joachims, editor. *Making Large-Scale Support Vector Machine Learning Practical*. MIT Press, Cambridge, MA, USA, 1998.

S. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606 – 617, 2007.

M. Kojima, N. Megiddo, T. Noma, and A. Yoshise. *A Unified Approach to Interior Point Algorithms for Linear Complementarity Problems*, volume 538 of *Lecture Notes in Computer Science*. Springer Verlag, Berlin, Germany, 1991.

C.J. Lin. On the convergence of the decomposition method for support vector machines. *IEEE Transactions on Neural Networks*, 12(12):1288 –1298, 2001.

H. Liu, S. Shah, and W. Jiang. On-line outlier detection and data cleaning. *Computers and Chemical Engineering*, 28:1635–1647, 2004.

S. Lucidi, L. Palagi, A. Risi, and M. Sciandrone. A convergent decomposition algorithm for support vector machines. *Comput. Optim. Appl.*, 38(2):217 –234, 2007.

D.J.C. MacKay. Bayesian interpolation. *Neural Computation*, 4:415–447, 1992.

D.J.C. Mackay. Bayesian non-linear modelling for the prediction competition. *ASHRAE Trans.*, 100(2):3704–3716, 1994.

A. Nemirovskii and Y. Nesterov. *Interior-Point Polynomial Algorithms in Convex Programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, USA, 1994.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Smoothed state estimates under abrupt changes using sum-of-norms regularization. *Automatica*, 48:595–605, 2012.

B. Oksendal. *Stochastic Differential Equations*. Springer, sixth edition, 2005.

J.A. Palmer, D.P. Wipf, K. Kreutz-Delgado, and B.D. Rao. Variational em algorithms for non-gaussian latent variable models. In *Proc. of NIPS*, 2006.

G. Pillonetto and B.M. Bell. Bayes and empirical Bayes semi-blind deconvolution using eigenfunctions of a prior covariance. *Automatica*, 43(10):1698–1712, 2007.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods: Support Vector Learning*, 1998.

M. Pontil and A. Verri. Properties of support vector machines. *Neural Computation*, 10:955–974, 1998.

C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

B. Ristic, S. Arulampalam, and N. Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House Publishers, 2004.

R.T. Rockafellar. *Convex Analysis*. Priceton Landmarks in Mathematics. Princeton University Press, 1970.

R.T. Rockafellar and R.J.B. Wets. *Variational Analysis*, volume 317. Springer, 1998.

S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11:305–345, 1999.

S. Saitoh. *Theory of Reproducing Kernels and Its Applications*. Longman, 1988.

H. H. Schaefer. *Topological Vector Spaces*. Springe-Verlag, 1970.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. (Adaptive Computation and Machine Learning). The MIT Press, 2001.

B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, 2000.

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. *Neural Networks and Computational Learning Theory*, 81:416–426, 2001.

A. J. Smola and B. Schölkopf. Bayesian kernel methods. In S. Mendelson and A. J. Smola, editors, *Machine Learning, Proceedings of the Summer School, Australian National University*, pages 65–117, Berlin, Germany, 2003. Springer-Verlag.

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

M. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

P. Tseng and S. Yun. A coordinate gradient descent method for linearly constrained smooth optimization and support vector machines training. *Comput. Optim. Appl.*, 47(2):1 –28, 2008.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, NY, USA, 1998.

G. Wahba. *Spline Models For Observational Data*. SIAM, Philadelphia, 1990.

G. Wahba. Support vector machines, reproducing kernel Hilbert spaces and randomized GACV. Technical Report 984, Department of Statistics, University of Wisconsin, 1998.

D.P. Wipf, B.D. Rao, and S. Nagarajan. Latent variable bayesian models for promoting sparsity. *IEEE Transactions on Information Theory*, 57:6236–6255, 2011.

S.J. Wright. *Primal-Dual Interior-Point Methods*. Siam, Englewood Cliffs, N.J., USA, 1997.

Y. Ye and K. Anstreicher. On quadratic and $o(\sqrt{nL})$ convergence of a predictor-corrector method for LCP. *Mathematical Programming*, 62(1-3):537–551, 1993.

E. H. Zarantonello. *Projections on Convex Sets in Hilbert Space and Spectral Theory*. Academic Press, 1971.

K. Zhang and J.T. Kwok. Clustered Nystrom method for large scale manifold learning and dimension reduction. *IEEE Transactions on Neural Networks*, 21(10):1576 –1587, 2010.

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

Journal of Machine Learning Research 14 (2013) 2729-2769 Submitted 10/12; Revised 4/13; Published 9/13

Improving CUR Matrix Decomposition and the Nyström Approximation via Adaptive Sampling

Shusen Wang WSS@ZJU.EDU.CN
College of Computer Science and Technology
Zhejiang University
Hangzhou, Zhejiang 310027, China

Zhihua Zhang* ZHIHUA@SJTU.EDU.CN
Department of Computer Science and Engineering
Shanghai Jiao Tong University
800 Dong Chuan Road, Shanghai, China 200240

Editor: Mehryar Mohri

Abstract

The CUR matrix decomposition and the Nyström approximation are two important low-rank matrix approximation techniques. The Nyström method approximates a symmetric positive semidefinite matrix in terms of a small number of its columns, while CUR approximates an arbitrary data matrix by a small number of its columns and rows. Thus, CUR decomposition can be regarded as an extension of the Nyström approximation.

In this paper we establish a more general error bound for the adaptive column/row sampling algorithm, based on which we propose more accurate CUR and Nyström algorithms with expected relative-error bounds. The proposed CUR and Nyström algorithms also have low time complexity and can avoid maintaining the whole data matrix in RAM. In addition, we give theoretical analysis for the lower error bounds of the standard Nyström method and the ensemble Nyström method. The main theoretical results established in this paper are novel, and our analysis makes no special assumption on the data matrices.

Keywords: large-scale matrix computation, CUR matrix decomposition, the Nyström method, randomized algorithms, adaptive sampling

1. Introduction

Large-scale matrices emerging from stocks, genomes, web documents, web images and videos everyday bring new challenges in modern data analysis. Most efforts have been focused on manipulating, understanding and interpreting large-scale data matrices. In many cases, matrix factorization methods are employed for constructing parsimonious and informative representations to facilitate computation and interpretation. A principled approach is the truncated singular value decomposition (SVD) which finds the best low-rank approximation of a data matrix. Applications of SVD such as eigenfaces (Sirovich and Kirby, 1987; Turk and Pentland, 1991) and latent semantic analysis (Deerwester et al., 1990) have been illustrated to be very successful.

∗. Corresponding author.

However, using SVD to find basis vectors and low-rank approximations has its limitations. As pointed out by Berry et al. (2005), it is often useful to find a low-rank matrix approximation which posses additional structures such as sparsity or nonnegativity. Since SVD or the standard QR decomposition for sparse matrices does not preserve sparsity in general, when the sparse matrix is large, computing or even storing such decompositions becomes challenging. Therefore it is useful to compute a low-rank matrix decomposition which preserves such structural properties of the original data matrix.

Another limitation of SVD is that the basis vectors resulting from SVD have little concrete meaning, which makes it very difficult for us to understand and interpret the data in question. An example of Drineas et al. (2008) and Mahoney and Drineas (2009) has well shown this viewpoint; that is, the vector $[(1/2)\text{age}-(1/\sqrt{2})\text{height}+(1/2)\text{income}]$, the sum of the significant uncorrelated features from a data set of people's features, is not particularly informative. Kuruvilla et al. (2002) have also claimed: "it would be interesting to try to find basis vectors for all experiment vectors, using actual experiment vectors and not artificial bases that offer little insight." Therefore, it is of great interest to represent a data matrix in terms of a small number of actual columns and/or actual rows of the matrix. *Matrix column selection* and the *CUR matrix decomposition* provide such techniques.

1.1 Matrix Column Selection

Column selection has been extensively studied in the theoretical computer science (TCS) and numerical linear algebra (NLA) communities. The work in TCS mainly focuses on choosing good columns by randomized algorithms with provable error bounds (Frieze et al., 2004; Deshpande et al., 2006; Drineas et al., 2008; Deshpande and Rademacher, 2010; Boutsidis et al., 2011; Guruswami and Sinop, 2012). The focus in NLA is then on deterministic algorithms, especially the rank-revealing QR factorizations, that select columns by pivoting rules (Foster, 1986; Chan, 1987; Stewart, 1999; Bischof and Hansen, 1991; Hong and Pan, 1992; Chandrasekaran and Ipsen, 1994; Gu and Eisenstat, 1996; Berry et al., 2005). In this paper we focus on randomized algorithms for column selection.

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, column selection algorithms aim to choose c columns of $\mathbf{A}$ to construct a matrix $\mathbf{C} \in \mathbb{R}^{m\times c}$ such that $\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\xi}$ achieves the minimum. Here "$\xi = 2$," "$\xi = F$," and "$\xi = *$" respectively represent the matrix spectral norm, the matrix Frobenius norm, and the matrix nuclear norm, and $\mathbf{C}^{\dagger}$ denotes the Moore-Penrose inverse of $\mathbf{C}$. Since there are $\binom{n}{c}$ possible choices of constructing $\mathbf{C}$, selecting the best subset is a hard problem.

In recent years, many polynomial-time approximate algorithms have been proposed. Among them we are especially interested in those algorithms with *multiplicative upper bounds*; that is, there exists a polynomial function $f(m,n,k,c)$ such that with c $(\geq k)$ columns selected from $\mathbf{A}$ the following inequality holds

$$\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\xi} \;\leq\; f(m,n,k,c)\,\|\mathbf{A}-\mathbf{A}_k\|_{\xi}$$

with high probability (w.h.p.) or in expectation w.r.t. $\mathbf{C}$. We call f the *approximation factor*. The bounds are strong when $f = 1+\varepsilon$ for an error parameter ε—they are known as *relative-error bounds*. Particularly, the bounds are called *constant-factor bounds* when f does not depend on m and n (Mahoney, 2011). The relative-error bounds and constant-factor bounds of the CUR matrix decomposition and the Nyström approximation are similarly defined.

However, the column selection method, also known as the $\mathbf{A} \approx \mathbf{CX}$ decomposition in some applications, has its limitations. For a large sparse matrix $\mathbf{A}$, its submatrix $\mathbf{C}$ is sparse, but the coefficient matrix $\mathbf{X} \in \mathbb{R}^{c \times n}$ is not sparse in general. The $\mathbf{CX}$ decomposition suffices when $m \gg n$, because $\mathbf{X}$ is small in size. However, when m and n are near equal, computing and storing the dense matrix $\mathbf{X}$ in RAM becomes infeasible. In such an occasion the CUR matrix decomposition is a very useful alternative.

1.2 The CUR Matrix Decomposition

The CUR matrix decomposition problem has been widely discussed in the literature (Goreinov et al., 1997a,b; Stewart, 1999; Tyrtyshnikov, 2000; Berry et al., 2005; Drineas and Mahoney, 2005; Mahoney et al., 2008; Bien et al., 2010), and it has been shown to be very useful in high dimensional data analysis. Particularly, a CUR decomposition algorithm seeks to find a subset of c columns of $\mathbf{A}$ to form a matrix $\mathbf{C} \in \mathbb{R}^{m \times c}$, a subset of r rows to form a matrix $\mathbf{R} \in \mathbb{R}^{r \times n}$, and an intersection matrix $\mathbf{U} \in \mathbb{R}^{c \times r}$ such that $\|\mathbf{A} - \mathbf{CUR}\|_\xi$ is small. Accordingly, we use $\tilde{\mathbf{A}} = \mathbf{CUR}$ to approximate $\mathbf{A}$.

Drineas et al. (2006) proposed a CUR algorithm with additive-error bound. Later on, Drineas et al. (2008) devised a randomized CUR algorithm which has relative-error bound w.h.p. if sufficiently many columns and rows are sampled. Mackey et al. (2011) established a divide-and-conquer method which solves the CUR problem in parallel. The CUR algorithms guaranteed by relative-error bounds are of great interest.

Unfortunately, the existing CUR algorithms usually require a large number of columns and rows to be chosen. For example, for an $m \times n$ matrix $\mathbf{A}$ and a target rank $k \ll \min\{m,n\}$, *the subspace sampling algorithm* (Drineas et al., 2008)—a classical CUR algorithm—requires $O(k\varepsilon^{-2}\log k)$ columns and $O(k\varepsilon^{-4}\log^2 k)$ rows to achieve relative-error bound w.h.p. The subspace sampling algorithm selects columns/rows according to the statistical leverage scores, so the computational cost of this algorithm is at least equal to the cost of the truncated SVD of $\mathbf{A}$, that is, $O(mnk)$ in general. However, maintaining a large scale matrix in RAM is often impractical, not to mention performing SVD. Recently, Drineas et al. (2012) devised fast approximation to statistical leverage scores which can be used to speedup the subspace sampling algorithm heuristically—yet no theoretical results have been reported that the leverage scores approximation can give provably efficient subspace sampling algorithm.

The CUR matrix decomposition problem has a close connection with the column selection problem. Especially, most CUR algorithms such as those of Drineas and Kannan (2003); Drineas et al. (2006, 2008) work in a two-stage manner where the first stage is a standard column selection procedure. Despite their strong resemblance, CUR is a harder problem than column selection because "one can get good columns or rows separately" does not mean that one can get good columns and rows together. If the second stage is naïvely solved by a column selection algorithm on $\mathbf{A}^T$, then the approximation factor will trivially be $\sqrt{2}f$[1] (Mahoney and Drineas, 2009). Thus, more sophisticated error analysis techniques for the second stage are indispensable in order to achieve relative-error bound.

1. It is because $\|\mathbf{A} - \mathbf{CUR}\|_F^2 = \|\mathbf{A} - \mathbf{CC}^\dagger\mathbf{A} + \mathbf{CC}^\dagger\mathbf{A} - \mathbf{CC}^\dagger\mathbf{AR}^\dagger\mathbf{R}\|_F^2 = \|(\mathbf{I} - \mathbf{CC}^\dagger)\mathbf{A}\|_F^2 + \|\mathbf{CC}^\dagger(\mathbf{A} - \mathbf{AR}^\dagger\mathbf{R})\|_F^2 \leq \|\mathbf{A} - \mathbf{CC}^\dagger\mathbf{A}\|_F^2 + \|\mathbf{A} - \mathbf{AR}^\dagger\mathbf{R}\|_F^2 \leq 2f^2\|\mathbf{A} - \mathbf{A}_k\|_F^2$, where the second equality follows from $(\mathbf{I} - \mathbf{CC}^\dagger)^T\mathbf{CC}^\dagger = 0$.

1.3 The Nyström Methods

The Nyström approximation is closely related to CUR, and it can potentially benefit from the advances in CUR techniques. Different from CUR, the Nyström methods are used for approximating symmetric positive semidefinite (SPSD) matrices. The methods approximate an SPSD matrix only using a subset of its columns, so they can alleviate computation and storage costs when the SPSD matrix in question is large in size. In fact, the Nyström methods have been extensively used in the machine learning community. For example, they have been applied to Gaussian processes (Williams and Seeger, 2001), kernel SVMs (Zhang et al., 2008), spectral clustering (Fowlkes et al., 2004), kernel PCA (Talwalkar et al., 2008; Zhang et al., 2008; Zhang and Kwok, 2010), etc.

The Nyström methods approximate any SPSD matrix in terms of a subset of its columns. Specifically, given an $m\times m$ SPSD matrix $\mathbf{A}$, they require sampling c ($< m$) columns of $\mathbf{A}$ to construct an $m\times c$ matrix $\mathbf{C}$. Since there exists an $m\times m$ permutation matrix Π such that $\Pi\mathbf{C}$ consists of the first c columns of $\Pi\mathbf{A}\Pi^T$, we always assume that $\mathbf{C}$ consists of the first c columns of $\mathbf{A}$ without loss of generality. We partition $\mathbf{A}$ and $\mathbf{C}$ as

$$\mathbf{A} = \begin{bmatrix} \mathbf{W} & \mathbf{A}_{21}^T \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \quad \text{and} \quad \mathbf{C} = \begin{bmatrix} \mathbf{W} \\ \mathbf{A}_{21} \end{bmatrix},$$

where $\mathbf{W}$ and $\mathbf{A}_{21}$ are of sizes $c\times c$ and $(m-c)\times c$, respectively. There are three models which are defined as follows.

- **The Standard Nyström Method**. The standard Nyström approximation to $\mathbf{A}$ is

$$\tilde{\mathbf{A}}_c^{\text{nys}} = \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T = \begin{bmatrix} \mathbf{W} & \mathbf{A}_{21}^T \\ \mathbf{A}_{21} & \mathbf{A}_{21}\mathbf{W}^\dagger\mathbf{A}_{21}^T \end{bmatrix}. \tag{1}$$

  Here $\mathbf{W}^\dagger$ is called the *intersection matrix*. The matrix $(\mathbf{W}_k)^\dagger$, where $k\le c$ and $\mathbf{W}_k$ is the best k-rank approximation to $\mathbf{W}$, is also used as an intersection matrix for constructing approximations with even lower rank. But using $\mathbf{W}^\dagger$ results in a tighter approximation than using $(\mathbf{W}_k)^\dagger$ usually.

- **The Ensemble Nyström Method** (Kumar et al., 2009). It selects a collection of t samples, each sample $\mathbf{C}^{(i)}$, $(i=1,\cdots,t)$, containing c columns of $\mathbf{A}$. Then the ensemble method combines the samples to construct an approximation in the form of

$$\tilde{\mathbf{A}}_{t,c}^{\text{ens}} = \sum_{i=1}^{t} \mu^{(i)}\mathbf{C}^{(i)}{\mathbf{W}^{(i)}}^\dagger{\mathbf{C}^{(i)}}^T, \tag{2}$$

  where $\mu^{(i)}$ are the weights of the samples. Typically, the ensemble Nyström method seeks to find out the weights by minimizing $\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{\text{ens}}\|_F$ or $\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{\text{ens}}\|_2$. A simple but effective strategy is to set the weights as $\mu^{(1)}=\cdots=\mu^{(t)}=\frac{1}{t}$.

- **The Modified Nyström Method** (proposed in this paper). It is defined as

$$\tilde{\mathbf{A}}_c^{\text{imp}} = \mathbf{C}\big(\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T\big)\mathbf{C}^T.$$

  This model is not strictly the Nyström method because it uses a quite different intersection matrix $\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T$. It costs $O(mc^2)$ time to compute the Moore-Penrose inverse $\mathbf{C}^\dagger$ and m^2c

flops to compute matrix multiplications. The matrix multiplications can be executed very efficiently in multi-processor environment, so ideally computing the intersection matrix costs time only linear in m. This model is more accurate (which will be justified in Section 4.3 and 4.4) but more costly than the conventional ones, so there is a trade-off between time and accuracy when deciding which model to use.

Here and later, we call those which use intersection matrix $\mathbf{W}^\dagger$ or $(\mathbf{W}_k)^\dagger$ *the conventional Nyström methods*, including the standard Nyström and the ensemble Nyström.

To generate effective approximations, much work has been built on the upper error bounds of the sampling techniques for the Nyström method. Most of the work, for example, Drineas and Mahoney (2005), Li et al. (2010), Kumar et al. (2009), Jin et al. (2011), and Kumar et al. (2012), studied the additive-error bound. With assumptions on matrix coherence, better additive-error bounds were obtained by Talwalkar and Rostamizadeh (2010), Jin et al. (2011), and Mackey et al. (2011). However, as stated by Mahoney (2011), additive-error bounds are less compelling than relative-error bounds. In one recent work, Gittens and Mahoney (2013) provided a relative-error bound for the first time, where the bound is in nuclear norm.

However, the error bounds of the previous Nyström methods are much weaker than those of the existing CUR algorithms, especially the relative-error bounds in which we are more interested (Mahoney, 2011). Actually, as will be proved in this paper, the lower error bounds of the standard Nyström method and the ensemble Nyström method are even much worse than the upper bounds of some existing CUR algorithms. This motivates us to improve the Nyström method by borrowing the techniques in CUR matrix decomposition.

1.4 Contributions and Outline

The main technical contribution of this work is the adaptive sampling bound in Theorem 5, which is an extension of Theorem 2.1 of Deshpande et al. (2006). Theorem 2.1 of Deshpande et al. (2006) bounds the error incurred by projection onto column or row space, while our Theorem 5 bounds the error incurred by the projection simultaneously onto column space and row space. We also show that Theorem 2.1 of Deshpande et al. (2006) can be regarded as a special case of Theorem 5.

More importantly, our adaptive sampling bound provides an approach for improving CUR and the Nyström approximation: no matter which relative-error column selection algorithm is employed, Theorem 5 ensures relative-error bounds for CUR and the Nyström approximation. We present the results in Corollary 7.

Based on the adaptive sampling bound in Theorem 5 and its corollary 7, we provide a concrete CUR algorithm which beats the best existing algorithm—the subspace sampling algorithm—both theoretically and empirically. The CUR algorithm is described in Algorithm 2 and analyzed in Theorem 8. In Table 1 we present a comparison between our proposed CUR algorithm and the subspace sampling algorithm. As we see, our algorithm requires much fewer columns and rows to achieve relative-error bound. Our method is more scalable for it works on only a few columns or rows of the data matrix in question; in contrast, the subspace sampling algorithm maintains the whole data matrix in RAM to implement SVD.

Another important application of the adaptive sampling bound is to yield an algorithm for the modified Nyström method. The algorithm has a strong relative-error upper bound: for a target rank k, by sampling $\frac{2k}{\varepsilon^2}\big(1+o(1)\big)$ columns it achieves relative-error bound in expectation. The results are shown in Theorem 10.

	#column (c)	**#row (r)**	**time**	**space**
Adaptive	$\frac{2k}{\varepsilon}(1+o(1))$	$\frac{c}{\varepsilon}(1+\varepsilon)$	Roughly $O(nk^2\varepsilon^{-4})+T_{\text{Multiply}}(mnk\varepsilon^{-1})$	$O(\max\{mc,nr\})$
Subspace	$O\left(\frac{k\log k}{\varepsilon^2}\right)$	$O\left(\frac{c\log c}{\varepsilon^2}\right)$	$O(mnk)$	$O(mn)$

Table 1: Comparisons between our *adaptive sampling* based CUR algorithm and the best existing algorithm—the *subspace sampling* algorithm of Drineas et al. (2008).

	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_F}{\max_{i,j}\|a_{ij}\|}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_2}{\max_{i,j}\|a_{ij}\|}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_*}{\max_{i,j}\|a_{ij}\|}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_F}{\|\mathbf{A}-\mathbf{A}_k\|_F}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_2}{\|\mathbf{A}-\mathbf{A}_k\|_2}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_*}{\|\mathbf{A}-\mathbf{A}_k\|_*}$
Standard	$\Omega\left(\frac{m\sqrt{k}}{c}\right)$	$\Omega\left(\frac{m}{c}\right)$	$\Omega(m-c)$	$\Omega\left(\sqrt{1+\frac{mk}{c^2}}\right)$	$\Omega\left(\frac{m}{c}\right)$	$\Omega\left(1+\frac{k}{c}\right)$
Ensemble	$\Omega\left(\frac{m\sqrt{k}}{c}\right)$	–	$\Omega(m-c)$	$\Omega\left(\sqrt{1+\frac{mk}{c^2}}\right)$	–	$\Omega\left(1+\frac{k}{c}\right)$

Table 2: Lower bounds of the standard Nyström method and the ensemble Nyström method. The blanks indicate the lower bounds are unknown to us. Here m denotes the column/row number of the SPSD matrix, c denotes the number of selected columns, and k denotes the target rank.

Finally, we establish a collection of lower error bounds of the standard Nyström and the ensemble Nyström that use $\mathbf{W}^\dagger$ as the intersection matrix. We show the lower bounds in Theorem 12 and Table 3; here Table 2 briefly summarizes the lower bounds in Table 3. From the table we can see that the upper error bound of our adaptive sampling algorithm for the modified Nyström method is even better than the lower bounds of the conventional Nyström methods.[2]

The remainder of the paper is organized as follows. In Section 2 we give the notation that will be used in this paper. In Section 3 we survey the previous work on the randomized column selection, CUR matrix decomposition, and Nyström approximation. In Section 4 we present our theoretical results and corresponding algorithms. In Section 5 we empirically evaluate our proposed CUR and Nyström algorithms. Finally, we conclude our work in Section 6. All proofs are deferred to the appendices.

2. Notation

First of all, we present the notation and notion that are used here and later. We let $\mathbf{I}_m$ denote the $m\times m$ identity matrix, $\mathbf{1}_m$ denote the $m\times 1$ vector of ones, and $\mathbf{0}$ denote a zero vector or matrix with appropriate size. For a matrix $\mathbf{A}=[a_{ij}]\in\mathbb{R}^{m\times n}$, we let $\mathbf{a}^{(i)}$ be its i-th row, $\mathbf{a}_j$ be its j-th column, and $\mathbf{A}_{i:j}$ be a submatrix consisting of its i to j-th columns ($i\le j$).

Let $\rho=\text{rank}(\mathbf{A})\le\min\{m,n\}$ and $k\le\rho$. The singular value decomposition (SVD) of $\mathbf{A}$ can be written as

$$\mathbf{A}=\sum_{i=1}^{\rho}\sigma_{\mathbf{A},i}\mathbf{u}_{\mathbf{A},i}\mathbf{v}_{\mathbf{A},i}^T=\mathbf{U}_\mathbf{A}\Sigma_\mathbf{A}\mathbf{V}_\mathbf{A}^T=\begin{bmatrix}\mathbf{U}_{\mathbf{A},k} & \mathbf{U}_{\mathbf{A},k\perp}\end{bmatrix}\begin{bmatrix}\Sigma_{\mathbf{A},k} & \mathbf{0}\\ \mathbf{0} & \Sigma_{\mathbf{A},k\perp}\end{bmatrix}\begin{bmatrix}\mathbf{V}_{\mathbf{A},k}^T\\ \mathbf{V}_{\mathbf{A},k\perp}^T\end{bmatrix},$$

2. This can be valid because the lower bounds in Table 2 do not hold when the intersection matrix is not $\mathbf{W}^\dagger$.

where $\mathbf{U}_{\mathbf{A},k}$ ($m\times k$), $\Sigma_{\mathbf{A},k}$ ($k\times k$), and $\mathbf{V}_{\mathbf{A},k}$ ($n\times k$) correspond to the top k singular values. We denote $\mathbf{A}_k = \mathbf{U}_{\mathbf{A},k}\Sigma_{\mathbf{A},k}\mathbf{V}_{\mathbf{A},k}^T$ which is the best (or closest) rank-k approximation to $\mathbf{A}$. We also use $\sigma_i(\mathbf{A}) = \sigma_{\mathbf{A},i}$ to denote the i-th largest singular value. When $\mathbf{A}$ is SPSD, the SVD is identical to the eigenvalue decomposition, in which case we have $\mathbf{U}_\mathbf{A} = \mathbf{V}_\mathbf{A}$.

We define the matrix norms as follows. Let $\|\mathbf{A}\|_1 = \sum_{i,j}|a_{ij}|$ be the ℓ_1-norm, $\|\mathbf{A}\|_F = (\sum_{i,j}a_{ij}^2)^{1/2} = (\sum_i \sigma_{\mathbf{A},i}^2)^{1/2}$ be the Frobenius norm, $\|\mathbf{A}\|_2 = \max_{\mathbf{x}\in\mathbb{R}^n,\|\mathbf{x}\|_2=1}\|\mathbf{A}\mathbf{x}\|_2 = \sigma_{\mathbf{A},1}$ be the spectral norm, and $\|\mathbf{A}\|_* = \sum_i \sigma_{\mathbf{A},i}$ be the nuclear norm. We always use $\|\cdot\|_\xi$ to represent $\|\cdot\|_2$, $\|\cdot\|_F$, or $\|\cdot\|_*$.

Based on SVD, the *statistical leverage scores* of the columns of $\mathbf{A}$ relative to the best rank-k approximation to $\mathbf{A}$ is defined as

$$\ell_j^{[k]} = \left\|\mathbf{v}_{\mathbf{A},k}^{(j)}\right\|_2^2, \quad j = 1,\cdots,n. \tag{3}$$

We have that $\sum_{j=1}^n \ell_j^{[k]} = k$. The leverage scores of the rows of $\mathbf{A}$ are defined according to $\mathbf{U}_{\mathbf{A},k}$. The leverage scores play an important role in low-rank matrix approximation. Informally speaking, the columns (or rows) with high leverage scores have greater influence in rank-k approximation than those with low leverage scores.

Additionally, let $\mathbf{A}^\dagger = \mathbf{V}_{\mathbf{A},\rho}\Sigma_{\mathbf{A},\rho}^{-1}\mathbf{U}_{\mathbf{A},\rho}^T$ be the Moore-Penrose inverse of $\mathbf{A}$ (Ben-Israel and Greville, 2003). When $\mathbf{A}$ is nonsingular, the Moore-Penrose inverse is identical to the matrix inverse. Given matrices $\mathbf{A}\in\mathbb{R}^{m\times n}$, $\mathbf{X}\in\mathbb{R}^{m\times p}$, and $\mathbf{Y}\in\mathbb{R}^{q\times n}$, $\mathbf{X}\mathbf{X}^\dagger\mathbf{A} = \mathbf{U}_\mathbf{X}\mathbf{U}_\mathbf{X}^T\mathbf{A}\in\mathbb{R}^{m\times n}$ is the projection of $\mathbf{A}$ onto the column space of $\mathbf{X}$, and $\mathbf{A}\mathbf{Y}^\dagger\mathbf{Y} = \mathbf{A}\mathbf{V}_\mathbf{Y}\mathbf{V}_\mathbf{Y}^T\in\mathbb{R}^{m\times n}$ is the projection of $\mathbf{A}$ onto the row space of $\mathbf{Y}$.

Finally, we discuss the computational costs of the matrix operations mentioned above. For an $m\times n$ general matrix $\mathbf{A}$ (assume $m\geq n$), it takes $O(mn^2)$ flops to compute the full SVD and $O(mnk)$ flops to compute the truncated SVD of rank k ($<n$). The computation of $\mathbf{A}^\dagger$ also takes $O(mn^2)$ flops. It is worth mentioning that, although multiplying an $m\times n$ matrix by an $n\times p$ matrix runs in mnp flops, it can be easily performed in parallel (Halko et al., 2011). In contrast, implementing operations like SVD and QR decomposition in parallel is much more difficult. So we denote the time complexity of such a matrix multiplication by $T_{\text{Multiply}}(mnp)$, which can be tremendously smaller than $O(mnp)$ in practice.

3. Previous Work

In Section 3.1 we present an adaptive sampling algorithm and its relative-error bound established by Deshpande et al. (2006). In Section 3.2 we highlight the near-optimal column selection algorithm of Boutsidis et al. (2011) which we will use in our CUR and Nyström algorithms for column/row sampling. In Section 3.3 we introduce two important CUR algorithms. In Section 3.4 we introduce the only known relative-error algorithm for the standard Nyström method.

3.1 The Adaptive Sampling Algorithm

Adaptive sampling is an effective and efficient column sampling algorithm for reducing the error incurred by the first round of sampling. After one has selected a small subset of columns (denoted $\mathbf{C}_1$), an adaptive sampling method is used to further select a proportion of columns according to the residual of the first round, that is, $\mathbf{A} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}$. The approximation error is guaranteed to be decreasing by a factor after the adaptive sampling (Deshpande et al., 2006). We show the result of Deshpande et al. (2006) in the following lemma.

Lemma 1 (The Adaptive Sampling Algorithm) (Deshpande et al., 2006) *Given a matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$*, we let* $\mathbf{C}_1 \in \mathbb{R}^{m\times c_1}$ *consist of* c_1 *columns of* $\mathbf{A}$*, and define the residual* $\mathbf{B} = \mathbf{A} - \mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{A}$*. Additionally, for* $i = 1, \cdots, n$*, we define*

$$p_i \;=\; \|\mathbf{b}_i\|_2^2 / \|\mathbf{B}\|_F^2.$$

We further sample c_2 *columns i.i.d. from* $\mathbf{A}$*, in each trial of which the i-th column is chosen with probability* p_i*. Let* $\mathbf{C}_2 \in \mathbb{R}^{m\times c_2}$ *contain the* c_2 *sampled columns and let* $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2] \in \mathbb{R}^{m\times(c_1+c_2)}$*. Then, for any integer* $k > 0$*, the following inequality holds:*

$$\mathbb{E}\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_F^2 \;\le\; \|\mathbf{A} - \mathbf{A}_k\|_F^2 + \frac{k}{c_2}\|\mathbf{A} - \mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{A}\|_F^2,$$

where the expectation is taken w.r.t. $\mathbf{C}_2$*.*

We will establish in Theorem 5 a more general and more useful error bound for this adaptive sampling algorithm. It can be shown that Lemma 1 is a special case of Theorem 5.

3.2 The Near-Optimal Column Selection Algorithm

Boutsidis et al. (2011) proposed a relative-error column selection algorithm which requires only $c = 2k\varepsilon^{-1}(1+o(1))$ columns get selected. Boutsidis et al. (2011) also proved the lower bound of the column selection problem which shows that no column selection algorithm can achieve relative-error bound by selecting less than $c = k\varepsilon^{-1}$ columns. Thus this algorithm is near optimal. Though an optimal algorithm recently proposed by Guruswami and Sinop (2012) attains the the lower bound, this algorithm is quite inefficient in comparison with the near-optimal algorithm. So we prefer to use the near-optimal algorithm in our CUR and Nyström algorithms for column/row sampling.

The near-optimal algorithm consists of three steps: the approximate SVD via random projection (Boutsidis et al., 2011; Halko et al., 2011), the dual set sparsification algorithm (Boutsidis et al., 2011), and the adaptive sampling algorithm (Deshpande et al., 2006). We describe the near-optimal algorithm in Algorithm 1 and present the theoretical analysis in Lemma 2.

Lemma 2 (The Near-Optimal Column Selection Algorithm) *Given a matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *of rank* ρ*, a target rank* k $(2 \le k < \rho)$*, and* $0 < \varepsilon < 1$*. Algorithm 1 selects*

$$c \;=\; \frac{2k}{\varepsilon}\Big(1 + o(1)\Big)$$

columns of $\mathbf{A}$ *to form a matrix* $\mathbf{C} \in \mathbb{R}^{m\times c}$*, then the following inequality holds:*

$$\mathbb{E}\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_F^2 \;\le\; (1+\varepsilon)\,\|\mathbf{A} - \mathbf{A}_k\|_F^2,$$

where the expectation is taken w.r.t. $\mathbf{C}$*. Furthermore, the matrix* $\mathbf{C}$ *can be obtained in* $O\big(mk^2\varepsilon^{-4/3} + nk^3\varepsilon^{-2/3}\big) + T_{\text{Multiply}}\big(mnk\varepsilon^{-2/3}\big)$ *time.*

This algorithm has the merits of low time complexity and space complexity. None of the three steps—the randomized SVD, the dual set sparsification algorithm, and the adaptive sampling—requires loading the whole of $\mathbf{A}$ into RAM. All of the three steps can work on only a small subset of the columns of $\mathbf{A}$. Though a relative-error algorithm recently proposed by Guruswami and Sinop (2012) requires even fewer columns, it is less efficient than the near-optimal algorithm.

Algorithm 1 The Near-Optimal Column Selection Algorithm of Boutsidis et al. (2011).

Input: a real matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, target rank k, error parameter $\varepsilon \in (0,1]$, target column number $c = \frac{2k}{\varepsilon}\big(1+o(1)\big)$;
Compute approximate truncated SVD via random projection such that $\mathbf{A}_k \approx \tilde{\mathbf{U}}_k\tilde{\Sigma}_k\tilde{\mathbf{V}}_k$;
Construct $\mathcal{U} \leftarrow$ columns of $(\mathbf{A}-\tilde{\mathbf{U}}_k\tilde{\Sigma}_k\tilde{\mathbf{V}}_k)$; $\mathcal{V} \leftarrow$ columns of $\tilde{\mathbf{V}}_k^T$;
Compute $\mathbf{s} \leftarrow$ Dual Set Spectral-Frobenius Sparsification Algorithm ($\mathcal{U}$, $\mathcal{V}$, $c-2k/\varepsilon$);
Construct $\mathbf{C}_1 \leftarrow \mathbf{A}\mathsf{Diag}(\mathbf{s})$, and then delete the all-zero columns;
Residual matrix $\mathbf{D} \leftarrow \mathbf{A}-\mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{A}$;
Compute sampling probabilities: $p_i = \|\mathbf{d}_i\|_2^2/\|\mathbf{D}\|_F^2$, $i=1,\cdots,n$;
Sampling $c_2 = 2k/\varepsilon$ columns from $\mathbf{A}$ with probability $\{p_1,\cdots,p_n\}$ to construct $\mathbf{C}_2$;
return $\mathbf{C} = [\mathbf{C}_1,\mathbf{C}_2]$.

3.3 Previous Work in CUR Matrix Decomposition

We introduce in this section two highly effective CUR algorithms: one is deterministic and the other is randomized.

3.3.1 THE SPARSE COLUMN-ROW APPROXIMATION (SCRA)

Stewart (1999) proposed a deterministic CUR algorithm and called it the sparse column-row approximation (SCRA). SCRA is based on the truncated pivoted QR decomposition via a quasi Gram-Schmidt algorithm. Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, the truncated pivoted QR decomposition procedure deterministically finds a set of columns $\mathbf{C} \in \mathbb{R}^{m\times c}$ by column pivoting, whose span approximates the column space of $\mathbf{A}$, and computes an upper triangular matrix $\mathbf{T}_\mathbf{C} \in \mathbb{R}^{c\times c}$ that orthogonalizes those columns. SCRA runs the same procedure again on $\mathbf{A}^T$ to select a set of rows $\mathbf{R} \in \mathbb{R}^{r\times n}$ and computes the corresponding upper triangular matrix $\mathbf{T}_\mathbf{R} \in \mathbb{R}^{r\times r}$. Let $\mathbf{C} = \mathbf{Q}_\mathbf{C}\mathbf{T}_\mathbf{C}$ and $\mathbf{R}^T = \mathbf{Q}_\mathbf{R}\mathbf{T}_\mathbf{R}$ denote the resulting truncated pivoted QR decomposition. The intersection matrix is computed by $\mathbf{U} = (\mathbf{T}_\mathbf{C}^T\mathbf{T}_\mathbf{C})^{-1}\mathbf{C}^T\mathbf{A}\mathbf{R}^T(\mathbf{T}_\mathbf{R}^T\mathbf{T}_\mathbf{R})^{-1}$. According to our experiments, this algorithm is quite effective but very time expensive, especially when c and r are large. Moreover, this algorithm does not have data-independent error bound.

3.3.2 THE SUBSPACE SAMPLING CUR ALGORITHM

Drineas et al. (2008) proposed a two-stage randomized CUR algorithm which has a relative-error bound with high probability (w.h.p.). In the first stage the algorithm samples c columns of $\mathbf{A}$ to construct $\mathbf{C}$, and in the second stage it samples r rows from $\mathbf{A}$ and $\mathbf{C}$ simultaneously to construct $\mathbf{R}$ and $\mathbf{W}$ and let $\mathbf{U} = \mathbf{W}^{\dagger}$. The sampling probabilities in the two stages are proportional to the leverage scores of $\mathbf{A}$ and $\mathbf{C}$, respectively. That is, in the first stage the sampling probabilities are proportional to the squared ℓ_2-norm of the rows of $\mathbf{V}_{\mathbf{A},k}$; in the second stage the sampling probabilities are proportional to the squared ℓ_2-norm of the rows of $\mathbf{U}_\mathbf{C}$. That is why it is called the *subspace sampling algorithm*. Here we show the main results of the subspace sampling algorithm in the following lemma.

Lemma 3 (Subspace Sampling for CUR) *Given an $m\times n$ matrix $\mathbf{A}$ and a target rank $k \ll \min\{m,n\}$, the subspace sampling algorithm selects $c = O(k\varepsilon^{-2}\log k\log(1/\delta))$ columns and $r = O\big(c\varepsilon^{-2}\log c\log(1/\delta)\big)$ rows without replacement. Then*

$$\|\mathbf{A}-\mathbf{C}\mathbf{U}\mathbf{R}\|_F \;=\; \big\|\mathbf{A}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{R}\big\|_F \;\le\; (1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_F,$$

holds with probability at least $1-\delta$*, where* $\mathbf{W}$ *contains the rows of* $\mathbf{C}$ *with scaling. The running time is dominated by the truncated SVD of* $\mathbf{A}$*, that is,* $O(mnk)$.

3.4 Previous Work in the Nyström Approximation

In a very recent work, Gittens and Mahoney (2013) established a framework for analyzing errors incurred by the standard Nyström method. Especially, the authors provided the first and the only known relative-error (in nuclear norm) algorithm for the standard Nyström method. The algorithm is described as follows and, its bound is shown in Lemma 4.

Like the CUR algorithm in Section 3.3.2, the Nyström algorithm also samples columns by the subspace sampling of Drineas et al. (2008). Each column is selected with probability $p_j = \frac{1}{k}\ell_j^{[k]}$ with replacement, where $\ell_1^{[k]}, \cdots, \ell_m^{[k]}$ are leverage scores defined in (3). After column sampling, $\mathbf{C}$ and $\mathbf{W}$ are obtained by scaling the selected columns, that is,

$$\mathbf{C} = \mathbf{A}(\mathbf{SD}) \quad \text{and} \quad \mathbf{W} = (\mathbf{SD})^T\mathbf{A}(\mathbf{SD}).$$

Here $\mathbf{S} \in \mathbb{R}^{m\times c}$ is a column selection matrix that $s_{ij} = 1$ if the i-th column of $\mathbf{A}$ is the j-th column selected, and $\mathbf{D} \in \mathbb{R}^{c\times c}$ is a diagonal scaling matrix satisfying $d_{jj} = \frac{1}{\sqrt{cp_i}}$ if $s_{ij} = 1$.

Lemma 4 (Subspace Sampling for the Nyström Approximation) *Given an* $m \times m$ *SPSD matrix* $\mathbf{A}$ *and a target rank* $k \ll m$*, the subspace sampling algorithm selects*

$$c = 3200\varepsilon^{-1}k\log(16k/\delta)$$

columns without replacement and constructs $\mathbf{C}$ *and* $\mathbf{W}$ *by scaling the selected columns. Then the inequality*

$$\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\right\|_* \leq (1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_*,$$

holds with probability at least $0.6-\delta$.

4. Main Results

We now present our main results. We establish a new error bound for the adaptive sampling algorithm in Section 4.1. We apply adaptive sampling to the CUR and modified Nyström problems, obtaining effective and efficient CUR and Nyström algorithms in Section 4.2 and Section 4.3 respectively. In Section 4.4 we study lower bounds of the conventional Nyström methods to demonstrate the advantages of our approach. Finally, in Section 4.5 we show that our expected bounds can extend to with high probability (w.h.p.) bounds.

4.1 Adaptive Sampling

The relative-error adaptive sampling algorithm is originally established in Theorem 2.1 of Deshpande et al. (2006) (see also Lemma 1 in Section 3.1). The algorithm is based on the following idea: after selecting a proportion of columns from $\mathbf{A}$ to form $\mathbf{C}_1$ by an arbitrary algorithm, the algorithm randomly samples additional c_2 columns according to the residual $\mathbf{A} - \mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{A}$. Here we prove a new and more general error bound for the same adaptive sampling algorithm.

Theorem 5 (The Adaptive Sampling Algorithm) *Given a matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *and a matrix* $\mathbf{C} \in \mathbb{R}^{m\times c}$ *such that* $\mathrm{rank}(\mathbf{C}) = \mathrm{rank}(\mathbf{C}\mathbf{C}^\dagger\mathbf{A}) = \rho$ $(\rho \le c \le n)$. *We let* $\mathbf{R}_1 \in \mathbb{R}^{r_1\times n}$ *consist of* r_1 *rows of* $\mathbf{A}$, *and define the residual* $\mathbf{B} = \mathbf{A} - \mathbf{A}\mathbf{R}_1^\dagger\mathbf{R}_1$. *Additionally, for* $i = 1,\cdots,m$, *we define*

$$p_i \;=\; \|\mathbf{b}^{(i)}\|_2^2/\|\mathbf{B}\|_F^2.$$

We further sample r_2 *rows i.i.d. from* $\mathbf{A}$, *in each trial of which the i-th row is chosen with probability* p_i. *Let* $\mathbf{R}_2 \in \mathbb{R}^{r_2\times n}$ *contain the* r_2 *sampled rows and let* $\mathbf{R} = [\mathbf{R}_1^T, \mathbf{R}_2^T]^T \in \mathbb{R}^{(r_1+r_2)\times n}$. *Then we have*

$$\mathbb{E}\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2 \;\le\; \|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_F^2 + \frac{\rho}{r_2}\|\mathbf{A} - \mathbf{A}\mathbf{R}_1^\dagger\mathbf{R}_1\|_F^2,$$

where the expectation is taken w.r.t. $\mathbf{R}_2$.

Remark 6 *This theorem shows a more general bound for adaptive sampling than the original one in Theorem 2.1 of Deshpande et al. (2006). The original one bounds the error incurred by projection onto the column space of* $\mathbf{C}$, *while Theorem 5 bounds the error incurred by projection onto the column space of* $\mathbf{C}$ *and row space of* $\mathbf{R}$ *simultaneously—such situation rises in problems such as CUR and the Nyström approximation. It is worth pointing out that Theorem 2.1 of Deshpande et al. (2006) is a direct corollary of this theorem when* $\mathbf{C} = \mathbf{A}_k$ *(i.e.,* $c = n$, $\rho = k$, *and* $\mathbf{C}\mathbf{C}^\dagger\mathbf{A} = \mathbf{A}_k$*).*

As discussed in Section 1.2, selecting good columns or rows separately does not ensure good columns and rows together for CUR and the Nyström approximation. Theorem 5 is thereby important for it guarantees the combined effect column and row selection. Guaranteed by Theorem 5, any column selection algorithm with relative-error bound can be applied to CUR and the Nyström approximation. We show the result in the following corollary.

Corollary 7 (Adaptive Sampling for CUR and the Nyström Approximation) *Given a matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$, *a target rank* k $(\ll m,n)$, *and a column selection algorithm* $\mathcal{A}_{col}$ *which achieves relative-error upper bound by selecting* $c \ge C(k,\varepsilon)$ *columns. Then we have the following results for CUR and the Nyström approximation.*

(1) *By selecting* $c \ge C(k,\varepsilon)$ *columns of* $\mathbf{A}$ *to construct* $\mathbf{C}$ *and* $r_1 = c$ *rows to construct* $\mathbf{R}_1$, *both using algorithm* $\mathcal{A}_{col}$, *followed by selecting additional* $r_2 = c/\varepsilon$ *rows using the adaptive sampling algorithm to construct* $\mathbf{R}_2$, *the CUR matrix decomposition achieves relative-error upper bound in expectation:*

$$\mathbb{E}\big\|\mathbf{A} - \mathbf{C}\mathbf{U}\mathbf{R}\big\|_F \;\le\; (1+\varepsilon)\big\|\mathbf{A} - \mathbf{A}_k\big\|_F,$$

where $\mathbf{R} = \big[\mathbf{R}_1^T, \mathbf{R}_2^T\big]^T$ *and* $\mathbf{U} = \mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger$.

(2) *Suppose* $\mathbf{A}$ *is an* $m\times m$ *symmetric matrix. By selecting* $c_1 \ge C(k,\varepsilon)$ *columns of* $\mathbf{A}$ *to construct* $\mathbf{C}_1$ *using* $\mathcal{A}_{col}$ *and selecting* $c_2 = c_1/\varepsilon$ *columns of* $\mathbf{A}$ *to construct* $\mathbf{C}_2$ *using the adaptive sampling algorithm, the modified Nyström method achieves relative-error upper bound in expectation:*

$$\mathbb{E}\big\|\mathbf{A} - \mathbf{C}\mathbf{U}\mathbf{C}^T\big\|_F \;\le\; (1+\varepsilon)\big\|\mathbf{A} - \mathbf{A}_k\big\|_F,$$

where $\mathbf{C} = \big[\mathbf{C}_1, \mathbf{C}_2\big]$ *and* $\mathbf{U} = \mathbf{C}^\dagger\mathbf{A}\big(\mathbf{C}^\dagger\big)^T$.

Based on Corollary 7, we attempt to solve CUR and the Nyström by adaptive sampling algorithms. We present concrete algorithms in Section 4.2 and 4.3.

Algorithm 2 Adaptive Sampling for CUR.

1: **Input:** a real matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, target rank k, $\varepsilon \in (0,1]$, target column number $c = \frac{2k}{\varepsilon}\big(1+o(1)\big)$, target row number $r = \frac{c}{\varepsilon}(1+\varepsilon)$;
2: Select $c = \frac{2k}{\varepsilon}\big(1+o(1)\big)$ columns of $\mathbf{A}$ to construct $\mathbf{C} \in \mathbb{R}^{m\times c}$ using Algorithm 1;
3: Select $r_1 = c$ rows of $\mathbf{A}$ to construct $\mathbf{R}_1 \in \mathbb{R}^{r_1\times n}$ using Algorithm 1;
4: Adaptively sample $r_2 = c/\varepsilon$ rows from $\mathbf{A}$ according to the residual $\mathbf{A} - \mathbf{A}\mathbf{R}_1^{\dagger}\mathbf{R}_1$;
5: **return** $\mathbf{C}$, $\mathbf{R} = [\mathbf{R}_1^T, \mathbf{R}_2^T]^T$, and $\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger}$.

4.2 Adaptive Sampling for CUR Matrix Decomposition

Guaranteed by the novel adaptive sampling bound in Theorem 5, we combine the near-optimal column selection algorithm of Boutsidis et al. (2011) and the adaptive sampling algorithm for solving the CUR problem, giving rise to an algorithm with a much tighter theoretical bound than existing algorithms. The algorithm is described in Algorithm 2 and its analysis is given in Theorem 8. Theorem 8 follows immediately from Lemma 2 and Corollary 7.

Theorem 8 (Adaptive Sampling for CUR) *Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and a positive integer $k \ll \min\{m,n\}$, the CUR algorithm described in Algorithm 2 randomly selects $c = \frac{2k}{\varepsilon}(1+o(1))$ columns of $\mathbf{A}$ to construct $\mathbf{C} \in \mathbb{R}^{m\times c}$, and then selects $r = \frac{c}{\varepsilon}(1+\varepsilon)$ rows of $\mathbf{A}$ to construct $\mathbf{R} \in \mathbb{R}^{r\times n}$. Then we have*

$$\mathbb{E}\|\mathbf{A} - \mathbf{C}\mathbf{U}\mathbf{R}\|_F \;=\; \mathbb{E}\|\mathbf{A} - \mathbf{C}(\mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger})\mathbf{R}\|_F \;\le\; (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_F.$$

The algorithm costs time $O\big((m+n)k^3\varepsilon^{-2/3} + mk^2\varepsilon^{-2} + nk^2\varepsilon^{-4}\big) + T_{\text{Multiply}}\big(mnk\varepsilon^{-1}\big)$ to compute matrices $\mathbf{C}$, $\mathbf{U}$ and $\mathbf{R}$.

When the algorithm is executed in a single-core processor, the time complexity of the CUR algorithm is linear in mn; when executed in multi-processor environment where matrix multiplication is performed in parallel, ideally the algorithm costs time only linear in $m+n$. Another advantage of this algorithm is that it avoids loading the whole $m\times n$ data matrix $\mathbf{A}$ into RAM. Neither the near-optimal column selection algorithm nor the adaptive sampling algorithm requires loading the whole of $\mathbf{A}$ into RAM. The most space-expensive operation throughout this algorithm is computation of the Moore-Penrose inverses of $\mathbf{C}$ and $\mathbf{R}$, which requires maintaining an $m\times c$ matrix or an $r\times n$ matrix in RAM. To compute the intersection matrix $\mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger}$, the algorithm needs to visit each entry of $\mathbf{A}$, but it is not RAM expensive because the multiplication can be done by computing $\mathbf{C}^{\dagger}\mathbf{a}_j$ for $j = 1,\cdots,n$ separately. The above analysis is also valid for the Nyström algorithm in Theorem 10.

Remark 9 *If we replace the near-optimal column selection algorithm in Theorem 8 by the optimal algorithm of Guruswami and Sinop (2012), it suffices to select $c = k\varepsilon^{-1}(1+o(1))$ columns and $r = c\varepsilon^{-1}(1+\varepsilon)$ rows totally. But the optimal algorithm is less efficient than the near-optimal algorithm.*

4.3 Adaptive Sampling for the Nyström Approximation

Theorem 5 provides an approach for bounding the approximation errors incurred by projection simultaneously onto column space and row space. Thus this approach can be applied to solve the modified Nyström method. The following theorem follows directly from Lemma 2 and Corollary 7.

	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_F}{\max_{i,j}\|a_{ij}\|}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_2}{\max_{i,j}\|a_{ij}\|}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_*}{\max_{i,j}\|a_{ij}\|}$
Standard	$0.99\sqrt{m-c-k+k\big(\frac{m+99k}{c+99k}\big)^2}$	$\frac{0.99(m+99)}{c+99}$	$0.99(m-c)\big(1+\frac{k}{c+99k}\big)$
Ensemble	$0.99\sqrt{(m-2c+\frac{c}{t}-k)+k\big(\frac{m-c+\frac{c}{t}+99k}{c+99k}\big)^2}$	–	$0.99(m-c)\big(1+\frac{k}{c+99k}\big)$

	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_F}{\|\mathbf{A}-\mathbf{A}_k\|_F}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_2}{\|\mathbf{A}-\mathbf{A}_k\|_2}$	$\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_*}{\|\mathbf{A}-\mathbf{A}_k\|_*}$
Standard	$\sqrt{1+\frac{m^2k-c^3}{c^2(m-k)}}$	$\frac{m}{c}$	$\frac{m-c}{m-k}\big(1+\frac{k}{c}\big)$
Ensemble	$\sqrt{\frac{m-2c+c/t-k}{m-k}\Big(1+\frac{k(m-2c+c/t)}{c^2}\Big)}$	–	$\frac{m-c}{m-k}\big(1+\frac{k}{c}\big)$

Table 3: Lower bounds of the standard Nyström method and the ensemble Nyström method. The blanks indicate the lower bounds are unknown to us. Here m denotes the column/row number of the SPSD matrix, c denotes the number of selected columns, and k denotes the target rank.

Theorem 10 (Adaptive Sampling for the Modified Nyström Method) *Given a symmetric matrix $\mathbf{A}\in\mathbb{R}^{m\times m}$ and a target rank k, with $c_1=\frac{2k}{\varepsilon}\big(1+o(1)\big)$ columns sampled by Algorithm 1 and $c_2=c_1/\varepsilon$ columns sampled by the adaptive sampling algorithm, that is, with totally $c=\frac{2k}{\varepsilon^2}\big(1+o(1)\big)$ columns being sampled, the approximation error incurred by the modified Nyström method is upper bounded by*

$$\mathbb{E}\big\|\mathbf{A}-\mathbf{C}\mathbf{U}\mathbf{C}^T\big\|_F \;\le\; \mathbb{E}\Big\|\mathbf{A}-\mathbf{C}\Big(\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T\Big)\mathbf{C}^T\Big\|_F \;\le\; (1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_F.$$

The algorithm costs time $O\big(mk^2\varepsilon^{-4}+mk^3\varepsilon^{-2/3}\big)+T_{\text{Multiply}}\big(m^2k\varepsilon^{-2}\big)$ in computing $\mathbf{C}$ and $\mathbf{U}$.

Remark 11 *The error bound in Theorem 10 is the only Frobenius norm relative-error bound for the Nyström approximation at present, and it is also a constant-factor bound. If one uses the optimal column selection algorithm of Guruswami and Sinop (2012), which is less efficient, the error bound is further improved: only $c=\frac{k}{\varepsilon^2}(1+o(1))$ columns are required. Furthermore, the theorem requires the matrix $\mathbf{A}$ to be symmetric, which is milder than the SPSD requirement made in the previous work.*

This is yet the strongest result for the Nyström approximation problem—much stronger than the best possible algorithms for the conventional Nyström method. We will illustrate this point by revealing the lower error bounds of the conventional Nyström methods.

4.4 Lower Error Bounds of the Conventional Nyström Methods

We now demonstrate to what an extent our modified Nyström method is superior over the conventional Nyström methods (namely the standard Nyström defined in (1) and the ensemble Nyström in (2)) by showing the lower error bounds of the conventional Nyström methods. The conventional Nyström methods work no better than the lower error bounds unless additional assumptions are made on the original matrix $\mathbf{A}$. We show in Theorem 12 the lower error bounds of the conventional Nyström methods; the results are briefly summarized previously in Table 2.

To derive lower error bounds, we construct two adversarial cases for the Nyström methods. To derive the spectral norm lower bounds, we use an SPSD matrix $\mathbf{B}$ whose diagonal entries equal to 1

and off-diagonal entries equal to $\alpha \in [0,1)$. For the Frobenius norm and nuclear norm bounds, we construct an $m \times m$ block diagonal matrix $\mathbf{A}$ which has k diagonal blocks, each of which is $\frac{m}{k} \times \frac{m}{k}$ in size and constructed in the same way as $\mathbf{B}$. For the lower bounds on $\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_\xi}{\max_{i,j}|a_{ij}|}$, α is set to be constant; for the bounds on $\frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_\xi}{\|\mathbf{A}-\mathbf{A}_k\|_\xi}$, α is set to be $\alpha \to 1$. The detailed proof of Theorem 12 is deferred to Appendix C.

Theorem 12 (Lower Error Bounds of the Nyström Methods) *Assume we are given an SPSD matrix $\mathbf{A} \in \mathbb{R}^{m\times m}$ and a target rank k. Let $\mathbf{A}_k$ denote the best rank-k approximation to $\mathbf{A}$. Let $\tilde{\mathbf{A}}$ denote either the rank-c approximation to $\mathbf{A}$ constructed by the standard Nyström method in (1), or the approximation constructed by the ensemble Nyström method in (2) with t non-overlapping samples, each of which contains c columns of $\mathbf{A}$. Then there exists an SPSD matrix such that for any sampling strategy the approximation errors of the conventional Nyström methods, that is, $\|\mathbf{A}-\tilde{\mathbf{A}}\|_\xi$, ($\xi = 2$, F, or "$*$"), are lower bounded by some factors which are shown in Table 3.*

Remark 13 *The lower bounds in Table 3 (or Table 2) show the conventional Nyström methods can be sometimes very ineffective. The spectral norm and Frobenius norm bounds even depend on m, so such bounds are not constant-factor bounds. Notice that the lower error bounds do not meet if $\mathbf{W}^\dagger$ is replaced by $\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T$, so our modified Nyström method is not limited by such lower bounds.*

4.5 Discussions of the Expected Relative-Error Bounds

The upper error bounds established in this paper all hold in expectation. Now we show that the expected error bounds immediately extend to w.h.p. bounds using Markov's inequality. Let the random variable $X = \|\mathbf{A}-\tilde{\mathbf{A}}\|_F / \|\mathbf{A}-\mathbf{A}_k\|_F$ denote the error ratio, where

$$\tilde{\mathbf{A}} = \mathbf{CUR} \text{ or } \mathbf{CUC}^T.$$

Then we have $\mathbb{E}(X) \le 1+\varepsilon$ by the preceding theorems. By applying Markov's inequality we have that

$$\mathbb{P}\big(X > 1+s\varepsilon\big) \;<\; \frac{\mathbb{E}(X)}{1+s\varepsilon} \;<\; \frac{1+\varepsilon}{1+s\varepsilon},$$

where s is an arbitrary constant greater than 1. Repeating the sampling procedure for t times and letting $X_{(i)}$ correspond to the error ratio of the i-th sample, we obtain an upper bound on the failure probability:

$$\mathbb{P}\Big(\min_i\{X_{(i)}\} > 1+s\varepsilon\Big) \;=\; \mathbb{P}\Big(X_{(i)} > 1+s\varepsilon\;\forall i = 1,\cdots,t\Big) \;<\; \Big(\frac{1+\varepsilon}{1+s\varepsilon}\Big)^t \;\triangleq\; \delta, \tag{4}$$

which decays exponentially with t. Therefore, by repeating the sampling procedure multiple times and choosing the best sample, our CUR and Nyström algorithms are also guaranteed with w.h.p. relative-error bounds. It follows directly from (4) that, by repeating the sampling procedure for

$$t \;\ge\; \frac{1+\varepsilon}{(s-1)\varepsilon}\log\Big(\frac{1}{\delta}\Big)$$

times, the inequality

$$\|\mathbf{A}-\tilde{\mathbf{A}}\|_F \;\le\; (1+s\varepsilon)\,\|\mathbf{A}-\mathbf{A}_k\|_F$$

holds with probability at least $1-\delta$.

For instance, we let $s = 1+\log(1/\delta)$, then by repeating the sampling procedure for $t \geq 1+1/\varepsilon$ times, the inequality

$$\|\mathbf{A}-\tilde{\mathbf{A}}\|_F \;\leq\; \Big(1+\varepsilon+\varepsilon\log(1/\delta)\Big)\,\|\mathbf{A}-\mathbf{A}_k\|_F$$

holds with probability at least $1-\delta$.

For another instance, we let $s = 2$, then by repeating the sampling procedure for $t \geq (1+1/\varepsilon)\log(1/\delta)$ times, the inequality

$$\|\mathbf{A}-\tilde{\mathbf{A}}\|_F \;\leq\; (1+2\varepsilon)\,\|\mathbf{A}-\mathbf{A}_k\|_F$$

holds with probability at least $1-\delta$.

5. Empirical Analysis

In Section 5.1 we empirical evaluate our CUR algorithms in comparison with the algorithms introduced in Section 3.3. In Section 5.2 we conduct empirical comparisons between the standard Nyström and our modified Nyström, and comparisons among three sampling algorithms. We report the approximation error incurred by each algorithm on each data set. The error ratio is defined by

$$\text{Error Ratio} \;=\; \frac{\|\mathbf{A}-\tilde{\mathbf{A}}\|_F}{\|\mathbf{A}-\mathbf{A}_k\|_F},$$

where $\tilde{\mathbf{A}} = \mathbf{CUR}$ for the CUR matrix decomposition, $\tilde{\mathbf{A}} = \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T$ for the standard Nyström method, and $\tilde{\mathbf{A}} = \mathbf{C}\big(\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}^{\dagger})^T\big)\mathbf{C}^T$ for the modified Nyström method.

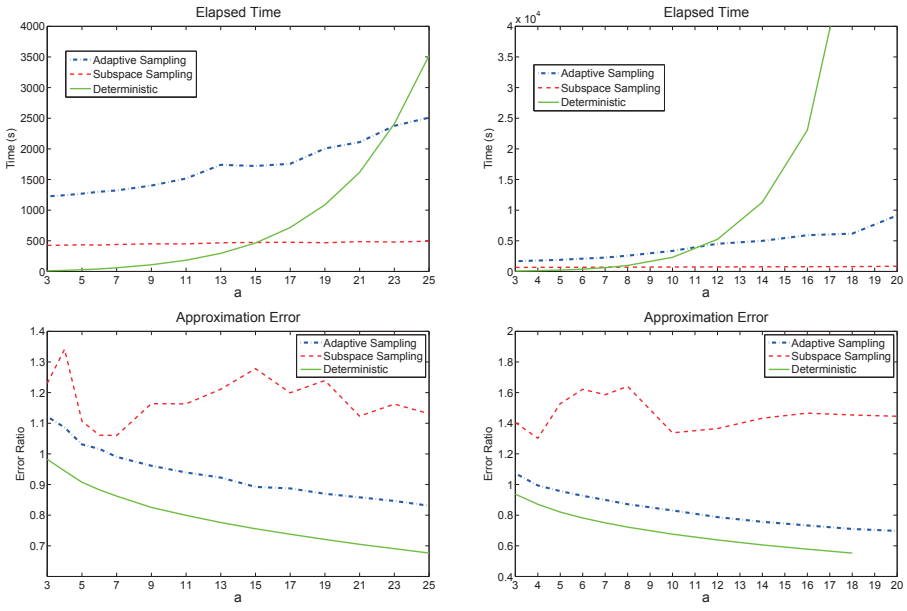
We conduct experiments on a workstation with two Intel Xeon 2.40GHz CPUs, 24GB RAM, and 64bit Windows Server 2008 system. We implement the algorithms in MATLAB R2011b, and use the MATLAB function 'svds' for truncated SVD. To compare the running time, all the computations are carried out in a single thread by setting 'maxNumCompThreads(1)' in MATLAB.

5.1 Comparison among the CUR Algorithms

In this section we empirically compare our adaptive sampling based CUR algorithm (Algorithm 2) with the subspace sampling algorithm of Drineas et al. (2008) and the deterministic sparse column-row approximation (SCRA) algorithm of Stewart (1999). For SCRA, we use the MATLAB code released by Stewart (1999). As for the subspace sampling algorithm, we compute the leverages scores exactly via the truncated SVD. Although the fast approximation to leverage scores (Drineas et al., 2012) can significantly speedup subspace sampling, we do not use it because the approximation has no theoretical guarantee when applied to subspace sampling.

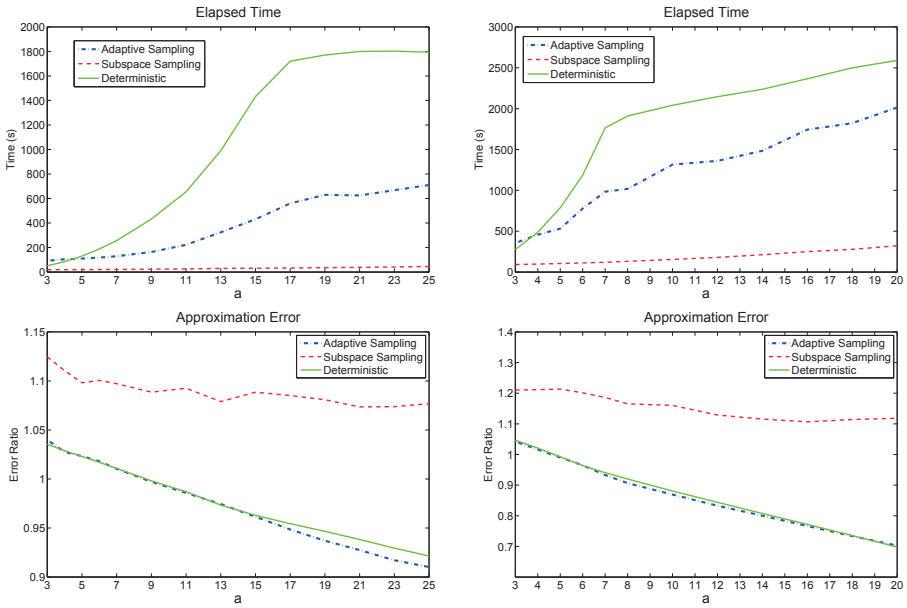
Data Set	Type	Size	#Nonzero Entries	Source
Enron Emails	text	$39,861 \times 28,102$	3,710,420	Bag-of-words, UCI
Dexter	text	$20,000 \times 2,600$	248,616	Guyon et al. (2004)
Farm Ads	text	$54,877 \times 4,143$	821,284	Mesterharm and Pazzani (2011)
Gisette	handwritten digit	$13,500 \times 5,000$	8,770,559	Guyon et al. (2004)

Table 4: A summary of the data sets for CUR matrix decomposition.



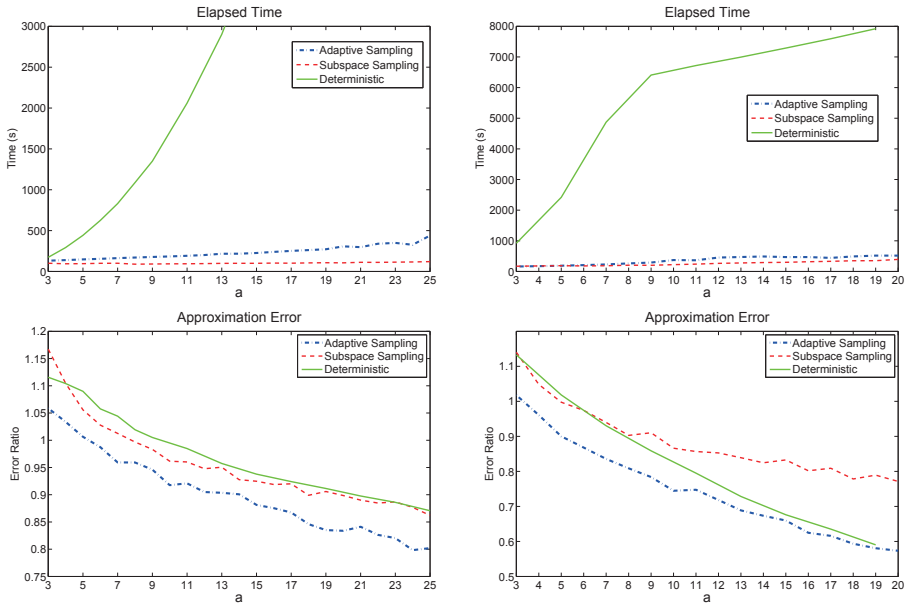
(a) $k = 10$, $c = ak$, and $r = ac$. (b) $k = 50$, $c = ak$, and $r = ac$.

Figure 1: Results of the CUR algorithms on the Enron data set.



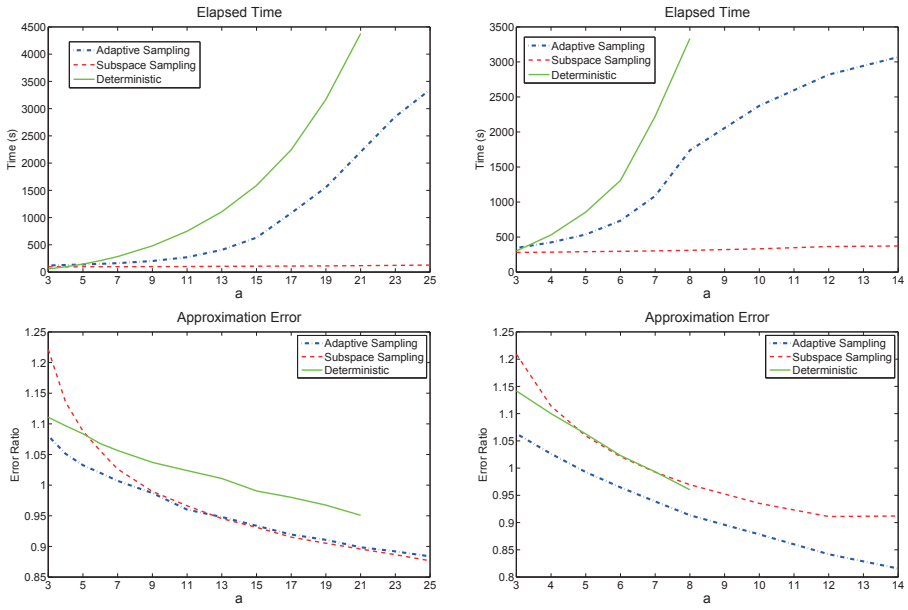
(a) $k = 10$, $c = ak$, and $r = ac$. (b) $k = 50$, $c = ak$, and $r = ac$.

Figure 2: Results of the CUR algorithms on the Dexter data set.



(a) $k = 10$, $c = ak$, and $r = ac$. (b) $k = 50$, $c = ak$, and $r = ac$.

Figure 3: Results of the CUR algorithms on the Farm Ads data set.



(a) $k = 10$, $c = ak$, and $r = ac$. (b) $k = 50$, $c = ak$, and $r = ac$.

Figure 4: Results of the CUR algorithms on the Gisette data set.

We conduct experiments on four UCI data sets (Frank and Asuncion, 2010) which are summarized in Table 4. Each data set is represented as a data matrix, upon which we apply the CUR algorithms. According to our analysis, the target rank k should be far less than m and n, and the column number c and row number r should be strictly greater than k. For each data set and each algorithm, we set $k = 10$ or 50, and $c = ak$, $r = ac$, where a ranges in each set of experiments. We repeat each of the two randomized algorithms 10 times, and report the minimum error ratio and the total elapsed time of the 10 rounds. We depict the error ratios and the elapsed time of the three CUR matrix decomposition algorithms in Figures 1, 2, 3, and 4.

We can see from Figures 1, 2, 3, and 4 that our adaptive sampling based CUR algorithm has much lower approximation error than the subspace sampling algorithm in all cases. Our adaptive sampling based algorithm is better than the deterministic SCRA on the Farm Ads data set and the Gisette data set, worse than SCRA on the Enron data set, and comparable to SCRA on the Dexter data set. In addition, the experimental results match our theoretical analysis in Section 4 very well. The empirical results all obey the theoretical relative-error upper bound

$$\frac{\|\mathbf{A}-\mathbf{CUR}\|_F}{\|\mathbf{A}-\mathbf{A}_k\|_F} \leq 1+\frac{2k}{c}\big(1+o(1)\big) = 1+\frac{2}{a}\big(1+o(1)\big).$$

As for the running time, the subspace sampling algorithm and our adaptive sampling based algorithm are much more efficient than SCRA, especially when c and r are large. Our adaptive sampling based algorithm is comparable to the subspace sampling algorithm when c and r are small; however, our algorithm becomes less efficient when c and r are large. This is due to the following reasons. First, the computational cost of the subspace sampling algorithm is dominated by the truncated SVD of $\mathbf{A}$, which is determined by the target rank k and the size and sparsity of the data matrix. However, the cost of our algorithm grows with c and r. Thus, our algorithm becomes less efficient when c and r are large. Second, the truncated SVD operation in MATLAB, that is, the 'svds' function, gains from sparsity, but our algorithm does not. The four data sets are all very sparse, so the subspace sampling algorithm has advantages. Third, the truncated SVD functions are very well implemented by MATLAB (not in MATLAB language but in Fortran/C). In contrast, our algorithm is implemented in MATLAB language, which is usually less efficient than Fortran/C.

5.2 Comparison among the Nyström Algorithms

In this section we empirically compare our adaptive sampling algorithm (in Theorem 10) with some other sampling algorithms including the subspace sampling of Drineas et al. (2008) and the uniform sampling, both without replacement. We also conduct comparison between the standard Nyström and our modified Nyström, both use the three sampling algorithms to select columns.

We test the algorithms on three data sets which are summarized in Table 5. The experiment setting follows Gittens and Mahoney (2013). For each data set we generate a radial basis function (RBF) kernel matrix $\mathbf{A}$ which is defined by

$$a_{ij} = \exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|_2^2}{2\sigma^2}\right),$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are data instances and σ is a scale parameter. Notice that the RBF kernel is dense in general. We set $\sigma = 0.2$ or 1 in our experiments. For each data set with different settings of σ, we fix a target rank $k = 10$, 20 or 50 and vary c in a very large range. We will discuss the choice

Data Set	#Instances	#Attributes	Source
Abalone	4,177	8	UCI (Frank and Asuncion, 2010)
Wine Quality	4,898	12	UCI (Cortez et al., 2009)
Letters	5,000	16	Statlog (Michie et al., 1994)

	$\|\mathbf{A}-\mathbf{A}_k\|_F/\|\mathbf{A}\|_F$			$\frac{m}{k}\mathsf{std}\left(\ell^{[k]}\right)$		
	$k=10$	$k=20$	$k=50$	$k=10$	$k=20$	$k=50$
Abalone ($\sigma=0.2$)	0.4689	0.3144	0.1812	0.8194	0.6717	0.4894
Abalone ($\sigma=1.0$)	0.0387	0.0122	0.0023	0.5879	0.8415	1.3830
Wine Quality ($\sigma=0.2$)	0.8463	0.7930	0.7086	1.8703	1.6490	1.3715
Wine Quality ($\sigma=1.0$)	0.0504	0.0245	0.0084	0.3052	0.5124	0.8067
Letters ($\sigma=0.2$)	0.9546	0.9324	0.8877	5.4929	3.9346	2.6210
Letters ($\sigma=1.0$)	0.1254	0.0735	0.0319	0.2481	0.2938	0.3833

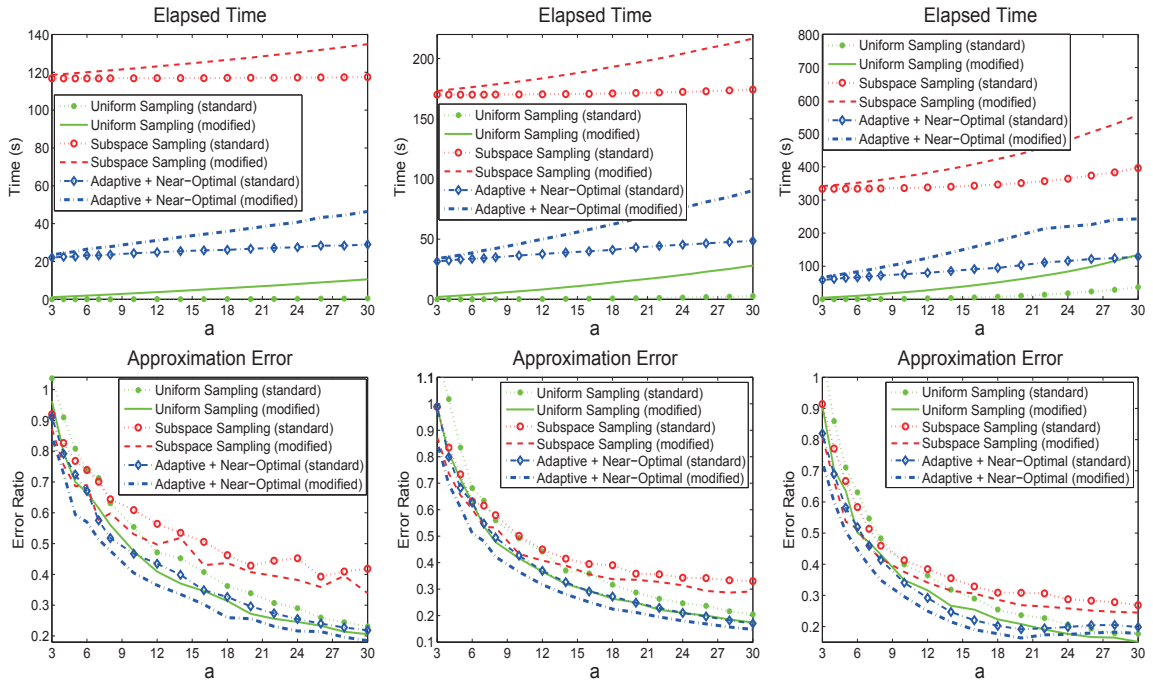
Table 5: A summary of the data sets for the Nyström approximation. In the second tabular $\mathsf{std}\left(\ell^{[k]}\right)$ denotes the standard deviation of the statistical leverage scores of $\mathbf{A}$ relative to the best rank-k approximation to $\mathbf{A}$. We use the normalization factor $\frac{m}{k}$ because $\frac{m}{k}\mathsf{mean}\left(\ell^{[k]}\right)=1$.

of σ and k in the following two paragraphs. We run each algorithm for 10 times, and report the the minimum error ratio as well as the total elapsed time of the 10 repeats. The results are shown in Figures 5, 6, and 7.

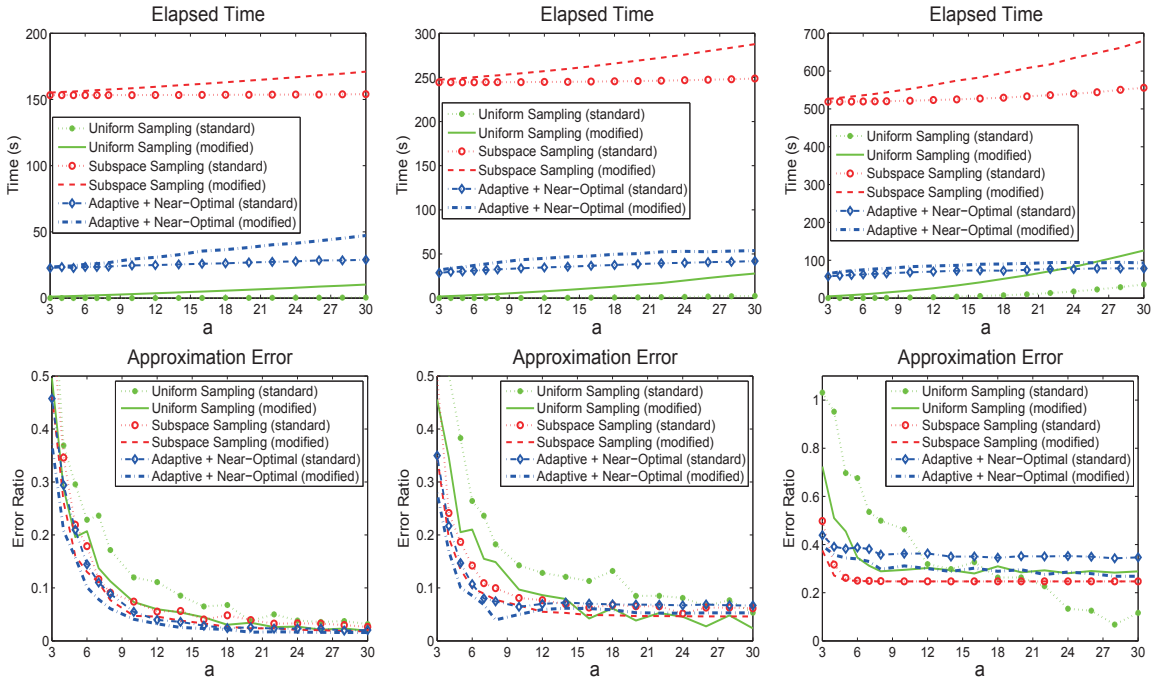
Table 5 provides useful implications on choosing the target rank k. In Table 5, $\frac{\|\mathbf{A}-\mathbf{A}_k\|_F}{\|\mathbf{A}\|_F}$ denotes ratio that is not captured by the best rank-k approximation to the RBF kernel, and the parameter σ has an influence on the ratio $\|\mathbf{A}-\mathbf{A}_k\|_F/\|\mathbf{A}\|_F$. When σ is large, the RBF kernel can be well approximated by a low-rank matrix, which implies that (i) a small k suffices when σ is large, and (ii) k should be set large when σ is small. So the settings ($\sigma=1$, $k=10$) and ($\sigma=0.2$, $k=50$) are more reasonable than the rest. Let us take the RBF kernel in the Abalone data set as an example. When $\sigma=1$, the rank-10 approximation well captures the kernel, so k can be safely set as small as 10; when $\sigma=0.2$, the target rank k should be set large, say larger than 50, otherwise the approximation is rough.

The standard deviation of the leverage scores reflects whether the advanced importance sampling techniques such as the subspace sampling and adaptive sampling are useful. Figures 5, 6, and 7 show that the advantage of the subspace sampling and adaptive sampling over the uniform sampling is significant whenever the standard deviation of the leverage scores is large (see Table 5), and vise versa. Actually, as reflected in Table 5, the parameter σ influences the homogeneity/heterogeneity of the leverage scores. Usually, when σ is small, the leverage scores become heterogeneous, and the effect of choosing "good" columns is significant.

The experimental results also show that the subspace sampling and adaptive sampling algorithms significantly outperform the uniform sampling when c is reasonably small, say $c<10k$. This indicates that the subspace sampling and adaptive sampling algorithms are good at choosing "good" columns as basis vectors. The effect is especially evident on the RBF kernel with the scale parameter $\sigma=0.2$, where the leverage scores are heterogeneous. In most cases our adaptive sampling algorithm achieves the lowest approximation error among the three algorithms. The error ratios of our adaptive sampling for the modified Nyström are in accordance with the theoretical bound in

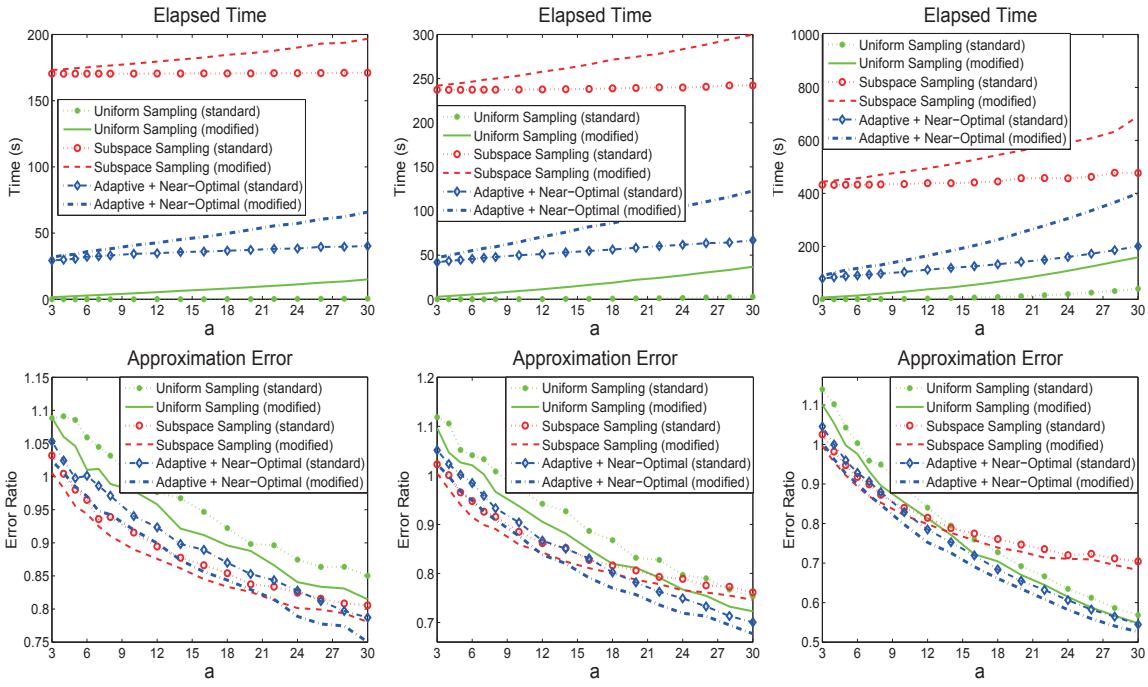


(a) $\sigma = 0.2$, $k = 10$, and $c = ak$. (b) $\sigma = 0.2$, $k = 20$, and $c = ak$. (c) $\sigma = 0.2$, $k = 50$, and $c = ak$.

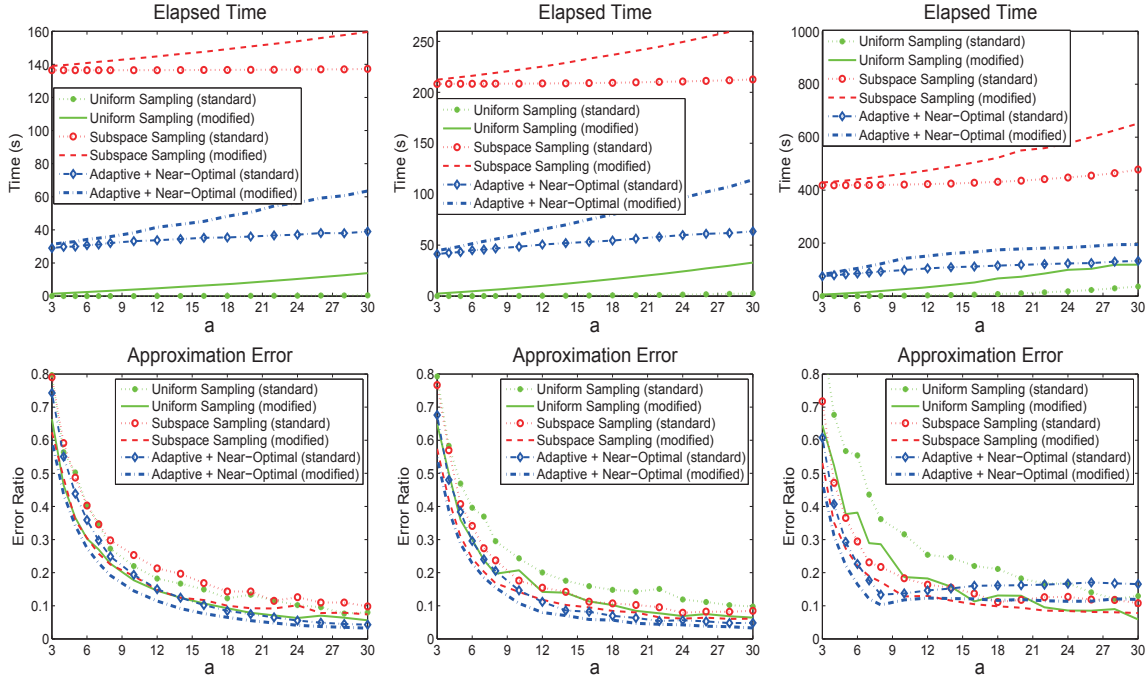


(d) $\sigma = 1$, $k = 10$, and $c = ak$. (e) $\sigma = 1$, $k = 20$, and $c = ak$. (f) $\sigma = 1$, $k = 50$, and $c = ak$.

Figure 5: Results of the Nyström algorithms on the RBF kernel in the Abalone data set.

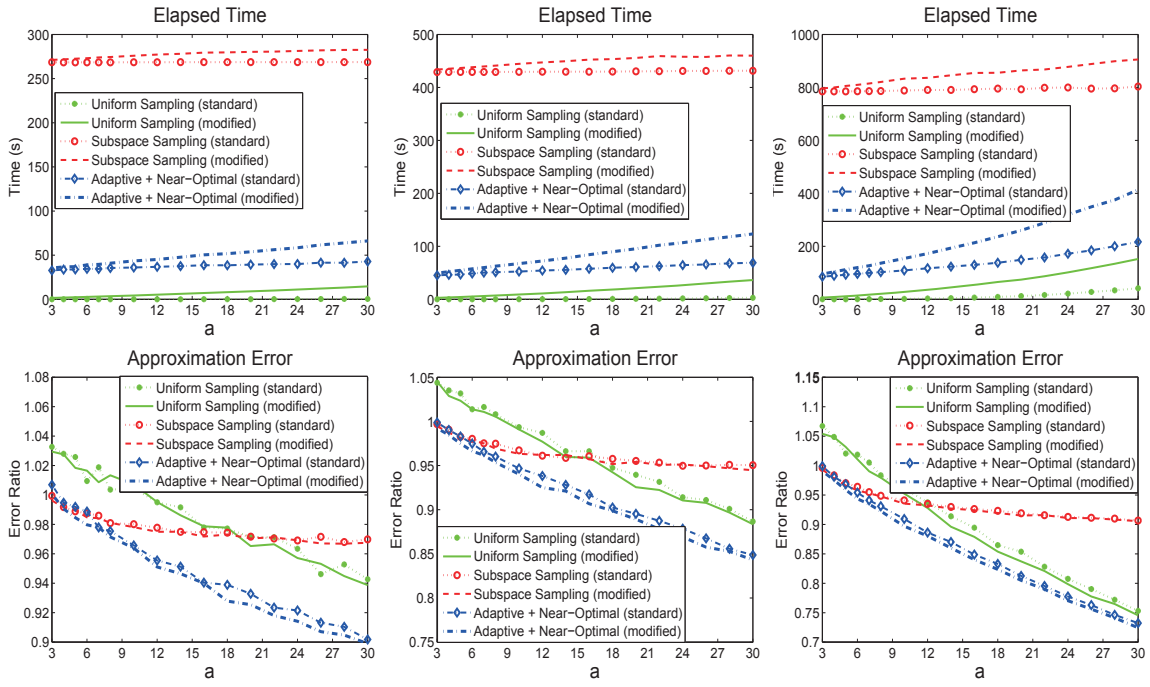


(a) $\sigma = 0.2$, $k = 10$, and $c = ak$. (b) $\sigma = 0.2$, $k = 20$, and $c = ak$. (c) $\sigma = 0.2$, $k = 50$, and $c = ak$.

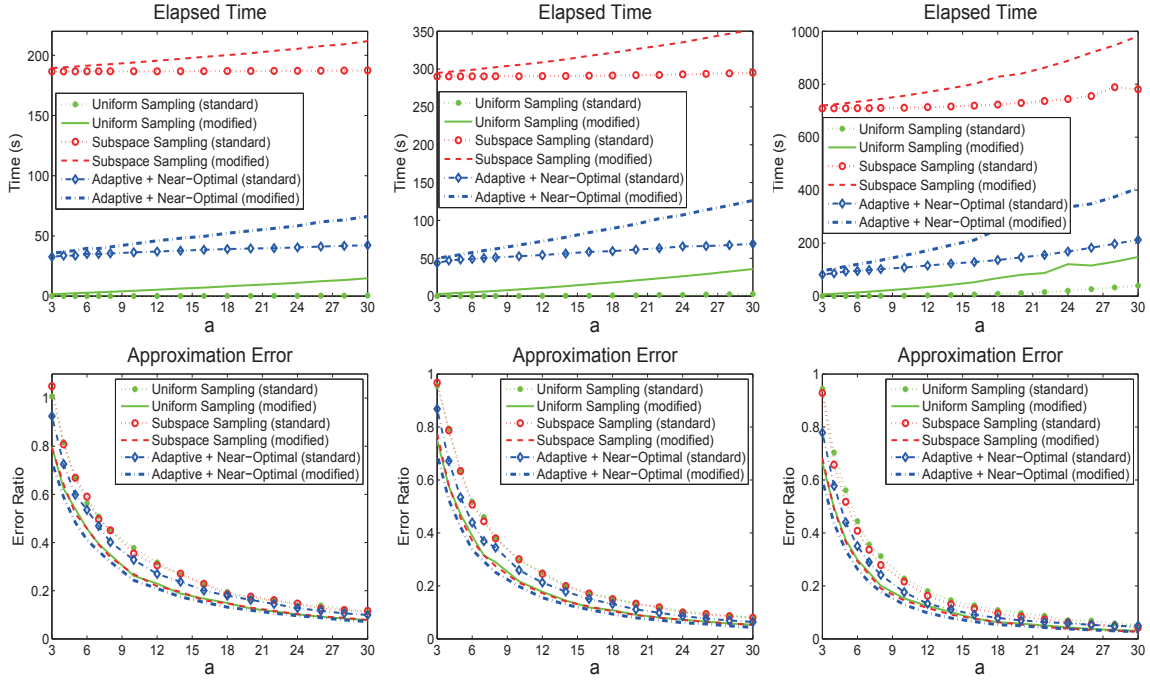


(d) $\sigma = 1$, $k = 10$, and $c = ak$. (e) $\sigma = 1$, $k = 20$, and $c = ak$. (f) $\sigma = 1$, $k = 50$, and $c = ak$.

Figure 6: Results of the Nyström algorithms on the RBF kernel in the Wine Quality data set.



(a) $\sigma = 0.2$, $k = 10$, and $c = ak$. (b) $\sigma = 0.2$, $k = 20$, and $c = ak$. (c) $\sigma = 0.2$, $k = 50$, and $c = ak$.



(d) $\sigma = 1$, $k = 10$, and $c = ak$. (e) $\sigma = 1$, $k = 20$, and $c = ak$. (f) $\sigma = 1$, $k = 50$, and $c = ak$.

Figure 7: Results of the Nyström algorithms on the RBF kernel in the Letters data set.

Theorem 10; that is,

$$\frac{\|\mathbf{A}-\mathbf{C}\mathbf{U}\mathbf{C}^T\|_F}{\|\mathbf{A}-\mathbf{A}_k\|_F} \leq 1+\sqrt{\frac{2k}{c}\big(1+o(1)\big)} = 1+\sqrt{\frac{2}{a}\big(1+o(1)\big)}.$$

As for the running time, our adaptive sampling algorithm is more efficient than the subspace sampling algorithm. This is partly because the RBF kernel matrix is dense, and hence the subspace sampling algorithm costs $O(m^2k)$ time to compute the truncated SVD.

Furthermore, the experimental results show that using $\mathbf{U}=\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T$ as the intersection matrix (denoted by "modified" in the figures) always leads to much lower error than using $\mathbf{U}=\mathbf{W}^\dagger$ (denoted by "standard"). However, our modified Nyström method costs more time to compute the intersection matrix than the standard Nyström method costs. Recall that the standard Nyström costs $O(c^3)$ time to compute $\mathbf{U}=\mathbf{W}^\dagger$ and that the modified Nyström costs $O(mc^2)+T_{\text{Multiply}}(m^2c)$ time to compute $\mathbf{U}=\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T$. So the users should make a trade-off between time and accuracy and decide whether it is worthwhile to sacrifice extra computational overhead for the improvement in accuracy by using the modified Nyström method.

6. Conclusion

In this paper we have built a novel and more general relative-error bound for the adaptive sampling algorithm. Accordingly, we have devised novel CUR matrix decomposition and Nyström approximation algorithms which demonstrate significant improvement over the classical counterparts. Our relative-error CUR algorithm requires only $c=2k\varepsilon^{-1}(1+o(1))$ columns and $r=c\varepsilon^{-1}(1+\varepsilon)$ rows selected from the original matrix. To achieve relative-error bound, the best previous algorithm—the subspace sampling algorithm—requires $c=O(k\varepsilon^{-2}\log k)$ columns and $r=O(c\varepsilon^{-2}\log c)$ rows. Our modified Nyström method is different from the conventional Nyström methods in that it uses a different intersection matrix. We have shown that our adaptive sampling algorithm for the modified Nyström achieves relative-error upper bound by sampling only $c=2k\varepsilon^{-2}(1+o(1))$ columns, which even beats the lower error bounds of the standard Nyström and the ensemble Nyström. Our proposed CUR and Nyström algorithms are scalable because they need only to maintain a small fraction of columns or rows in RAM, and their time complexities are low provided that matrix multiplication can be highly efficiently executed. Finally, the empirical comparison has also demonstrated the effectiveness and efficiency of our algorithms.

Acknowledgments

This work has been supported in part by the Natural Science Foundations of China (No. 61070239) and the Scholarship Award for Excellent Doctoral Student granted by Chinese Ministry of Education.

Appendix A. The Dual Set Sparsification Algorithm

For the sake of self-contained, we attach the dual set sparsification algorithm and describe some implementation details. The deterministic dual set sparsification algorithm is established by Boutsidis et al. (2011) and severs as an important step in the near-optimal column selection algorithm (described in Lemma 2 and Algorithm 1 in this paper). We show the dual set sparsification algorithm

Algorithm 3 Deterministic Dual Set Spectral-Frobenius Sparsification Algorithm.

1: **Input:** $\mathcal{U} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^l$, $(l < n)$; $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^n \subset \mathbb{R}^k$, with $\sum_{i=1}^n \mathbf{v}_i\mathbf{v}_i^T = \mathbf{I}_k$ $(k < n)$; $k < r < n$;
2: **Initialize:** $\mathbf{s}_0 = \mathbf{0}$, $\mathbf{A}_0 = \mathbf{0}$;
3: Compute $\|\mathbf{x}_i\|_2^2$ for $i = 1, \cdots, n$, and then compute $\delta_U = \frac{\sum_{i=1}^n \|\mathbf{x}_i\|_2^2}{1-\sqrt{k/r}}$;
4: **for** $\tau = 0$ to $r-1$ **do**
5: Compute the eigenvalue decomposition of $\mathbf{A}_\tau$;
6: Find any index j in $\{1, \cdots, n\}$ and compute a weight $t > 0$ such that

$$\delta_U^{-1}\|\mathbf{x}_j\|_2^2 \;\le\; t^{-1} \;\le\; \frac{\mathbf{v}_j^T\Big(\mathbf{A}_\tau - (L_\tau+1)\mathbf{I}_k\Big)^{-2}\mathbf{v}_j}{\phi(L_\tau+1, \mathbf{A}_\tau) - \phi(L_\tau, \mathbf{A}_\tau)} - \mathbf{v}_j^T\Big(\mathbf{A}_\tau - (L_\tau+1)\mathbf{I}_k\Big)^{-1}\mathbf{v}_j;$$

where

$$\phi(L, \mathbf{A}) = \sum_{i=1}^k \Big(\lambda_i(\mathbf{A}) - L\Big)^{-1}, \qquad L_\tau = \tau - \sqrt{rk};$$

7: Update the j-th component of $\mathbf{s}_\tau$ and $\mathbf{A}_\tau$: $\mathbf{s}_{\tau+1}[j] = \mathbf{s}_\tau[j] + t$, $\mathbf{A}_{\tau+1} = \mathbf{A}_\tau + t\mathbf{v}_j\mathbf{v}_j^T$;
8: **end for**
9: **return** $\mathbf{s} = \frac{1-\sqrt{k/r}}{r}\mathbf{s}_r$.

algorithm in Algorithm 3 and its bounds in Lemma 14, and we also analyze the time complexity using our defined notation.

Lemma 14 (Dual Set Spectral-Frobenius Sparsification) *Let $\mathcal{U} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathbb{R}^l$ $(l < n)$ contain the columns of an arbitrary matrix $\mathbf{X} \in \mathbb{R}^{l\times n}$. Let $\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\} \subset \mathbb{R}^k$ $(k < n)$ be a decompositions of the identity, that is, $\sum_{i=1}^n \mathbf{v}_i\mathbf{v}_i^T = \mathbf{I}_k$. Given an integer r with $k < r < n$, Algorithm 3 deterministically computes a set of weights $s_i \ge 0$ $(i = 1, \cdots, n)$ at most r of which are non-zero, such that*

$$\lambda_k\Big(\sum_{i=1}^n s_i\mathbf{v}_i\mathbf{v}_i^T\Big) \ge \Big(1 - \sqrt{\frac{k}{r}}\Big)^2 \qquad \textit{and} \qquad \mathrm{tr}\Big(\sum_{i=1}^n s_i\mathbf{x}_i\mathbf{x}_i^T\Big) \le \|\mathbf{X}\|_F^2.$$

The weights s_i can be computed deterministically in $O\big(rnk^2\big) + T_{\text{Multiply}}\big(nl\big)$ time.

Here we mention some implementation issues of Algorithm 3 which were not described in detail by Boutsidis et al. (2011). In each iteration the algorithm performs once eigenvalue decomposition: $\mathbf{A}_\tau = \mathbf{W}\Lambda\mathbf{W}^T$. Here $\mathbf{A}_\tau$ is guaranteed to be SPSD in each iteration. Since

$$\Big(\mathbf{A}_\tau - \alpha\mathbf{I}_k\Big)^q \;=\; \mathbf{W}\mathrm{Diag}\Big((\lambda_1-\alpha)^q, \cdots, (\lambda_k-\alpha)^q\Big)\mathbf{W}^T,$$

$(\mathbf{A}_\tau - (L_\tau+1)\mathbf{I}_k)^q$ can be efficiently computed based on the eigenvalue decomposition of $\mathbf{A}_\tau$. With the eigenvalues at hand, $\phi(L, \mathbf{A}_\tau)$ can also be computed directly.

The algorithm runs in r iterations. In each iteration, the eigenvalue decomposition of $\mathbf{A}_\tau$ requires $O(k^3)$, and the n comparisons in Line 6 each requires $O(k^2)$. Moreover, computing $\|\mathbf{x}_i\|_2^2$ for each $\mathbf{x}_i$ requires $T_{\text{Multiply}}(nl)$. Overall, the running time of Algorithm 3 is at most $O(rk^3) + O(rnk^2) + T_{\text{Multiply}}(nl) = O(rnk^2) + T_{\text{Multiply}}(nl)$.

The near-optimal column selection algorithm described in Lemma 2 has three steps: randomized SVD via random projection which costs $O\big(mk^2\varepsilon^{-4/3}\big) + T_{\text{Multiply}}\big(mnk\varepsilon^{-2/3}\big)$ time, the dual

set sparsification algorithm which costs $O\big(nk^3\varepsilon^{-2/3}\big)+T_{\text{Multiply}}\big(mn\big)$ time, and the adaptive sampling algorithm which costs $O\big(mk^2\varepsilon^{-4/3}\big)+T_{\text{Multiply}}\big(mnk\varepsilon^{-2/3}\big)$ time. Therefore, the near-optimal column selection algorithm costs totally $O\big(mk^2\varepsilon^{-4/3}+nk^3\varepsilon^{-2/3}\big)+T_{\text{Multiply}}\big(mnk\varepsilon^{-2/3}\big)$ time.

Appendix B. Proofs of the Adaptive Sampling Bounds

We present the proofs of Theorem 5, Corollary 7, Theorem 8, and Theorem 10 in Appendices B.1, B.2, B.3, and B.4, respectively.

B.1 The Proof of Theorem 5

Theorem 5 can be equivalently expressed in Theorem 15. In order to stick to the column space convention throughout this paper, we prove Theorem 15 instead of Theorem 5.

Theorem 15 (The Adaptive Sampling Algorithm) *Given a matrix $\mathbf{A}\in\mathbb{R}^{m\times n}$ and a matrix $\mathbf{R}\in\mathbb{R}^{r\times n}$ such that $\text{rank}(\mathbf{R})=\text{rank}(\mathbf{AR}^\dagger\mathbf{R})=\rho$ ($\rho\le r\le m$), let $\mathbf{C}_1\in\mathbb{R}^{m\times c_1}$ consist of c_1 columns of $\mathbf{A}$, and define the residual $\mathbf{B}=\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}$. For $i=1,\cdots,n$, let*

$$p_i \;=\; \|\mathbf{b}_i\|_2^2/\|\mathbf{B}\|_F^2,$$

where $\mathbf{b}_i$ is the i-th column of the matrix $\mathbf{B}$. Sample further c_2 columns from $\mathbf{A}$ in c_2 i.i.d. trials, where in each trial the i-th column is chosen with probability p_i. Let $\mathbf{C}_2\in\mathbb{R}^{m\times c_2}$ contain the c_2 sampled columns and $\mathbf{C}=[\mathbf{C}_1,\mathbf{C}_2]\in\mathbb{R}^{m\times(c_1+c_2)}$ contain the columns of both $\mathbf{C}_1$ and $\mathbf{C}_2$, all of which are columns of $\mathbf{A}$. Then the following inequality holds:

$$\mathbb{E}\|\mathbf{A}-\mathbf{CC}^\dagger\mathbf{AR}^\dagger\mathbf{R}\|_F^2 \;\le\; \|\mathbf{A}-\mathbf{AR}^\dagger\mathbf{R}\|_F^2+\frac{\rho}{c_2}\|\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\|_F^2.$$

where the expectation is taken w.r.t. $\mathbf{C}_2$.

Proof With a little abuse of symbols, we use bold uppercase letters to denote random matrices and bold lowercase to denote random vectors, without distinguishing between random matrices/vectors and non-random matrices/vectors.

We denote the j-th column of $\mathbf{V}_{\mathbf{AR}^\dagger\mathbf{R},\rho}\in\mathbb{R}^{n\times\rho}$ as $\mathbf{v}_j$, and the (i,j)-th entry of $\mathbf{V}_{\mathbf{AR}^\dagger\mathbf{R},\rho}$ as v_{ij}. Define random vectors $\mathbf{x}_{j,(l)}\in\mathbb{R}^m$ such that for $j=1,\cdots,n$ and $l=1,\cdots,c_2$,

$$\mathbf{x}_{j,(l)}=\frac{v_{ij}}{p_i}\mathbf{b}_i=\frac{v_{ij}}{p_i}\Big(\mathbf{a}_i-\mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{a}_i\Big)\quad\text{with probability } p_i,\quad\text{for } i=1,\cdots,n,$$

Notice that $\mathbf{x}_{j,(l)}$ is a linear function of a column of $\mathbf{A}$ sampled from the above defined distribution. We have that

$$\begin{aligned}
\mathbb{E}[\mathbf{x}_{j,(l)}] &= \sum_{i=1}^n p_i\frac{v_{ij}}{p_i}\mathbf{b}_i \;=\; \mathbf{B}\mathbf{v}_j,\\
\mathbb{E}\|\mathbf{x}_{j,(l)}\|_2^2 &= \sum_{i=1}^n p_i\frac{v_{ij}^2}{p_i^2}\|\mathbf{b}_i\|_2^2 \;=\; \sum_{i=1}^n\frac{v_{ij}^2}{\|\mathbf{b}_i\|_2^2/\|\mathbf{B}\|_F^2}\|\mathbf{b}_i\|_2^2 \;=\; \|\mathbf{B}\|_F^2.
\end{aligned}$$

Then we let $\mathbf{x}_j = \frac{1}{c_2}\sum_{l=1}^{c_2}\mathbf{x}_{j,(l)}$, we have

$$\begin{aligned}\mathbb{E}[\mathbf{x}_j] &= \mathbb{E}[\mathbf{x}_{j,(l)}] = \mathbf{B}\mathbf{v}_j,\\ \mathbb{E}\|\mathbf{x}_j-\mathbf{B}\mathbf{v}_j\|_2^2 &= \mathbb{E}\Big\|\mathbf{x}_j-\mathbb{E}[\mathbf{x}_j]\Big\|_2^2 = \frac{1}{c_2}\mathbb{E}\Big\|\mathbf{x}_{j,(l)}-\mathbb{E}[\mathbf{x}_{j,(l)}]\Big\|_2^2 = \frac{1}{c_2}\mathbb{E}\|\mathbf{x}_{j,(l)}-\mathbf{B}\mathbf{v}_j\|_2^2.\end{aligned}$$

According to the construction of $\mathbf{x}_1,\cdots,\mathbf{x}_\rho$, we define the c_2 columns of $\mathbf{A}$ to be $\mathbf{C}_2\in\mathbb{R}^{m\times c_2}$. Note that all the random vectors $\mathbf{x}_1\cdots,\mathbf{x}_\rho$ lie in the subspace $\mathrm{span}(\mathbf{C}_1)+\mathrm{span}(\mathbf{C}_2)$. We define random vectors

$$\mathbf{w}_j = \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\mathbf{v}_j+\mathbf{x}_j = \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\mathbf{v}_j+\mathbf{x}_j,\qquad \text{for } j=1,\cdots,\rho,$$

where the second equality follows from Lemma 16; that is, $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\mathbf{v}_j=\mathbf{A}\mathbf{v}_j$ if $\mathbf{v}_j$ is one of the top ρ right singular vectors of $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$. Then we have that any set of random vectors $\{\mathbf{w}_1,\cdots,\mathbf{w}_\rho\}$ lies in $\mathrm{span}(\mathbf{C})=\mathrm{span}(\mathbf{C}_1)+\mathrm{span}(\mathbf{C}_2)$. Let $\mathbf{W}=[\mathbf{w}_1,\cdots,\mathbf{w}_\rho]$ be a random matrix, we have that $\mathrm{span}(\mathbf{W})\subset\mathrm{span}(\mathbf{C})$. The expectation of $\mathbf{w}_j$ is

$$\mathbb{E}[\mathbf{w}_j] = \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\mathbf{v}_j+\mathbb{E}[\mathbf{x}_j] = \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\mathbf{v}_j+\mathbf{B}\mathbf{v}_j = \mathbf{A}\mathbf{v}_j,$$

therefore we have that

$$\mathbf{w}_j-\mathbf{A}\mathbf{v}_j = \mathbf{x}_j-\mathbf{B}\mathbf{v}_j.$$

The expectation of $\|\mathbf{w}_j-\mathbf{A}\mathbf{v}_j\|_2^2$ is

$$\begin{aligned}\mathbb{E}\|\mathbf{w}_j-\mathbf{A}\mathbf{v}_j\|_2^2 &= \mathbb{E}\|\mathbf{x}_j-\mathbf{B}\mathbf{v}_j\|_2^2 = \frac{1}{c_2}\mathbb{E}\|\mathbf{x}_{j,(l)}-\mathbf{B}\mathbf{v}_j\|_2^2\\ &= \frac{1}{c_2}\mathbb{E}\|\mathbf{x}_{j,(l)}\|_2^2-\frac{2}{c_2}(\mathbf{B}\mathbf{v}_j)^T\mathbb{E}[\mathbf{x}_{j,(l)}]+\frac{1}{c_2}\|\mathbf{B}\mathbf{v}_j\|_2^2\\ &= \frac{1}{c_2}\mathbb{E}\|\mathbf{x}_{j,(l)}\|_2^2-\frac{1}{c_2}\|\mathbf{B}\mathbf{v}_j\|_2^2 = \frac{1}{c_2}\|\mathbf{B}\|_F^2-\frac{1}{c_2}\|\mathbf{B}\mathbf{v}_j\|_2^2\\ &\le \frac{1}{c_2}\|\mathbf{B}\|_F^2.\end{aligned}\tag{5}$$

To complete the proof, we denote

$$\mathbf{F} = (\sum_{q=1}^{\rho}\sigma_q^{-1}\mathbf{w}_q\mathbf{u}_q^T)\mathbf{A}\mathbf{R}^\dagger\mathbf{R},$$

where σ_q is the q-th largest singular value of $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$ and $\mathbf{u}_q$ is the corresponding left singular vector of $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$. The column space of $\mathbf{F}$ is contained in $\mathrm{span}(\mathbf{W})$ ($\subset\mathrm{span}(\mathbf{C})$), and thus

$$\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2\le\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{W}\mathbf{W}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2\le\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{F}\|_F^2.$$

We use $\mathbf{F}$ to bound the error $\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2$. That is,

$$\begin{aligned}\mathbb{E}\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2 &= \mathbb{E}\|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}+\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2\\ &= \mathbb{E}\Big[\|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2\Big]\\ &\le \|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\mathbb{E}\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{F}\|_F^2,\end{aligned}\tag{6}$$

where (6) is due to that $\mathbf{A}(\mathbf{I}-\mathbf{R}^\dagger\mathbf{R})$ is orthogonal to $(\mathbf{I}-\mathbf{C}\mathbf{C}^\dagger)\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$. Since $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$ and $\mathbf{F}$ both lie on the space spanned by the right singular vectors of $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$ (i.e., $\{\mathbf{v}_j\}_{j=1}^{\rho}$), we decompose $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{F}$ along $\{\mathbf{v}_j\}_{j=1}^{\rho}$, obtaining that

$$\begin{aligned}
\mathbb{E}\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2 &\leq \|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\mathbb{E}\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{F}\|_F^2,\\
&= \|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\sum_{j=1}^{\rho}\mathbb{E}\left\|(\mathbf{A}\mathbf{R}^\dagger\mathbf{R}-\mathbf{F})\mathbf{v}_j\right\|_2^2\\
&= \|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\sum_{j=1}^{\rho}\mathbb{E}\left\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\mathbf{v}_j-(\sum_{q=1}^{\rho}\sigma_q^{-1}\mathbf{w}_q\mathbf{u}_q^T)\sigma_j\mathbf{u}_j\right\|_2^2\\
&= \|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\sum_{j=1}^{\rho}\mathbb{E}\left\|\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\mathbf{v}_j-\mathbf{w}_j\right\|_2^2\\
&= \|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\sum_{j=1}^{\rho}\mathbb{E}\|\mathbf{A}\mathbf{v}_j-\mathbf{w}_j\|_2^2 \qquad (7)\\
&\leq \|\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\|_F^2+\frac{\rho}{c_2}\|\mathbf{B}\|_F^2, \qquad (8)
\end{aligned}$$

where (7) follows from Lemma 16 and (8) follows from (5). ■

Lemma 16 *We are given a matrix* $\mathbf{A}\in\mathbb{R}^{m\times n}$ *and a matrix* $\mathbf{R}\in\mathbb{R}^{r\times n}$ *such that* $\mathrm{rank}(\mathbf{A}\mathbf{R}^\dagger\mathbf{R})=\mathrm{rank}(\mathbf{R})=\rho$ $(\rho\leq r\leq m)$. *Letting* $\mathbf{v}_j\in\mathbb{R}^n$ *be the* j*-th top right singular vector of* $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$*, we have that*

$$\mathbf{A}\mathbf{R}^\dagger\mathbf{R}\mathbf{v}_j \;=\; \mathbf{A}\mathbf{v}_j, \qquad \textit{for } j=1,\cdots,\rho.$$

Proof First let $\mathbf{V}_{\mathbf{R},\rho}\in\mathbb{R}^{n\times\rho}$ contain the top ρ right singular vectors of $\mathbf{R}$. Then the projection of $\mathbf{A}$ onto the row space of $\mathbf{R}$ is $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}=\mathbf{A}\mathbf{V}_{\mathbf{R},\rho}\mathbf{V}_{\mathbf{R},\rho}^T$. Let the thin SVD of $\mathbf{A}\mathbf{V}_{\mathbf{R},\rho}\in\mathbb{R}^{m\times\rho}$ be $\tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{V}}^T$, where $\tilde{\mathbf{V}}\in\mathbb{R}^{\rho\times\rho}$. Then the compact SVD of $\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$ is

$$\mathbf{A}\mathbf{R}^\dagger\mathbf{R} \;=\; \mathbf{A}\mathbf{V}_{\mathbf{R},\rho}\mathbf{V}_{\mathbf{R},\rho}^T \;=\; \tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{V}}^T\mathbf{V}_{\mathbf{R},\rho}^T.$$

According to the definition, $\mathbf{v}_j$ is the j-th column of $(\mathbf{V}_{\mathbf{R},\rho}\tilde{\mathbf{V}})\in\mathbb{R}^{n\times\rho}$. Thus $\mathbf{v}_j$ lies on the column space of $\mathbf{V}_{\mathbf{R},\rho}$, and $\mathbf{v}_j$ is orthogonal to $\mathbf{V}_{\mathbf{R},\rho\perp}$. Finally, since $\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}=\mathbf{A}\mathbf{V}_{\mathbf{R},\rho\perp}\mathbf{V}_{\mathbf{R},\rho\perp}^T$, we have that $\mathbf{v}_j$ is orthogonal to $\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R}$, that is, $(\mathbf{A}-\mathbf{A}\mathbf{R}^\dagger\mathbf{R})\mathbf{v}_j=\mathbf{0}$, which directly proves the lemma. ■

B.2 The Proof of Corollary 7

Since $\mathbf{C}$ is constructed by columns of $\mathbf{A}$ and the column space of $\mathbf{C}$ is contained in the column space of $\mathbf{A}$, we have $\mathrm{rank}(\mathbf{C}\mathbf{C}^\dagger\mathbf{A})=\mathrm{rank}(\mathbf{C})=\rho\leq c$. Consequently, the assumptions of Theorem 5 are satisfied. The assumptions in turn imply

$$\begin{aligned}
\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_F &\leq (1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_F,\\
\|\mathbf{A}-\mathbf{A}\mathbf{R}_1^\dagger\mathbf{R}_1\|_F &\leq (1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_F,
\end{aligned}$$

and $c/r_2 = \varepsilon$. It then follows from Theorem 5 that

$$\begin{aligned}
\mathbb{E}_{\mathbf{R}}\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2 &= \mathbb{E}_{\mathbf{R}_1}\left[\mathbb{E}_{\mathbf{R}_2}\left[\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\|_F^2\Big|\mathbf{R}_1\right]\right] \\
&\leq \mathbb{E}_{\mathbf{R}_1}\left[\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_F^2+\frac{\rho}{r_2}\|\mathbf{A}-\mathbf{A}\mathbf{R}_1^{\dagger}\mathbf{R}_1\|_F^2\right] \\
&\leq \|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_F^2+\frac{c}{r_2}(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_F^2 \\
&= \|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_F^2+\varepsilon(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_F^2.
\end{aligned}$$

Furthermore, we have that

$$\begin{aligned}
\left[\mathbb{E}\|\mathbf{A}-\mathbf{C}\mathbf{U}\mathbf{R}\|_F\right]^2 &\leq \mathbb{E}\|\mathbf{A}-\mathbf{C}\mathbf{U}\mathbf{R}\|_F^2 = \mathbb{E}_{\mathbf{C},\mathbf{R}}\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\|_F^2 \\
&= \mathbb{E}_{\mathbf{C}}\left[\mathbb{E}_{\mathbf{R}}\left[\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\|_F^2\Big|\mathbf{C}\right]\right] \\
&\leq \mathbb{E}_{\mathbf{C}}\left[\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_F^2+\varepsilon(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_F^2\right] \\
&\leq (1+\varepsilon)^2\|\mathbf{A}-\mathbf{A}_k\|_k^2,
\end{aligned}$$

which yields the error bound for CUR matrix decomposition.

When the matrix $\mathbf{A}$ is symmetric, the matrix $\mathbf{C}_1^T$ consists of the rows $\mathbf{A}$, and thus we can use Theorem 15 (which is identical to Theorem 5) to prove the error bound for the Nyström approximation. By replacing $\mathbf{R}$ in Theorem 15 by $\mathbf{C}_1^T$, we have that

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}_1^{\dagger})^T\mathbf{C}_1^T\right\|_F^2 &\leq \left\|\mathbf{A}-\mathbf{A}(\mathbf{C}_1^{\dagger})^T\mathbf{C}_1^T\right\|_F^2+\frac{c_1}{c_2}\left\|\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{A}\right\|_F^2 \\
&= \left(1+\frac{c_1}{c_2}\right)\left\|\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{A}\right\|_F^2,
\end{aligned}$$

where the expectation is taken w.r.t. $\mathbf{C}_2$. Together with the inequality

$$\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}^{\dagger})^T\mathbf{C}^T\right\|_F^2 \leq \left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}_1^{\dagger})^T\mathbf{C}_1^T\right\|_F^2$$

given by Lemma 17, we have that

$$\begin{aligned}
\mathbb{E}_{\mathbf{C}_1,\mathbf{C}_2}\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}^{\dagger})^T\mathbf{C}^T\right\|_F^2 &\leq \mathbb{E}_{\mathbf{C}_1,\mathbf{C}_2}\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}_1^{\dagger})^T\mathbf{C}_1^T\right\|_F^2 \\
&= \left(1+\frac{c_1}{c_2}\right)\mathbb{E}_{\mathbf{C}_1}\left\|\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^{\dagger}\mathbf{A}\right\|_F^2 \\
&= (1+\varepsilon)^2\left\|\mathbf{A}-\mathbf{A}_k\right\|_F^2.
\end{aligned}$$

Hence $\mathbb{E}\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}^{\dagger})^T\mathbf{C}^T\right\|_F \leq \left[\mathbb{E}\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}^{\dagger})^T\mathbf{C}^T\right\|_F^2\right]^{-\frac{1}{2}} \leq (1+\varepsilon)\left\|\mathbf{A}-\mathbf{A}_k\right\|_F$.

Lemma 17 *Given an $m\times m$ matrix $\mathbf{A}$ and an $m\times c$ matrix $\mathbf{C}=[\mathbf{C}_1,\mathbf{C}_2]$, the following inequality holds:*

$$\left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}^{\dagger})^T\mathbf{C}^T\right\|_F^2 \leq \left\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}_1^{\dagger})^T\mathbf{C}_1^T\right\|_F^2.$$

Proof Let $\mathcal{P}_{\mathbf{C}}\mathbf{A} = \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}$ denote the projection of $\mathbf{A}$ onto the column space of $\mathbf{C}$, and $\bar{\mathcal{P}}_{\mathbf{C}} = \mathbf{I}_m - \mathbf{C}\mathbf{C}^{\dagger}$ denote the projector onto the space orthogonal to the column space of $\mathbf{C}$. It has been shown by Halko et al. (2011) that, for any matrix $\mathbf{A}$, if $\mathrm{span}(\mathbf{M}) \subset \mathrm{span}(\mathbf{N})$, then the following inequalities hold:

$$\|\mathcal{P}_{\mathbf{M}}\mathbf{A}\|_{\xi} \leq \|\mathcal{P}_{\mathbf{N}}\mathbf{A}\|_{\xi} \quad \text{and} \quad \|\bar{\mathcal{P}}_{\mathbf{M}}\mathbf{A}\|_{\xi} \geq \|\bar{\mathcal{P}}_{\mathbf{N}}\mathbf{A}\|_{\xi}.$$

Accordingly, $\mathbf{A}\mathcal{P}^T_{\mathbf{R}^T} = \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}$ is the projection of $\mathbf{A}$ onto the row space of $\mathbf{R} \in \mathbb{R}^{r\times n}$. We further have that

$$\begin{aligned}\|\mathbf{A} - \mathcal{P}_{\mathbf{C}}\mathbf{A}\mathcal{P}^T_{\mathbf{C}}\|_F^2 &= \|\mathbf{A} - \mathcal{P}_{\mathbf{C}}\mathbf{A} + \mathcal{P}_{\mathbf{C}}\mathbf{A} - \mathcal{P}_{\mathbf{C}}\mathbf{A}\mathcal{P}^T_{\mathbf{C}}\|_F^2 \\ &= \|\bar{\mathcal{P}}_{\mathbf{C}}\mathbf{A} + \mathcal{P}_{\mathbf{C}}\mathbf{A}\bar{\mathcal{P}}^T_{\mathbf{C}}\|_F^2 \;=\; \|\bar{\mathcal{P}}_{\mathbf{C}}\mathbf{A}\|_F^2 + \|\mathcal{P}_{\mathbf{C}}\mathbf{A}\bar{\mathcal{P}}^T_{\mathbf{C}}\|_F^2\end{aligned}$$

and

$$\begin{aligned}\|\mathbf{A} - \mathcal{P}_{\mathbf{C}}\mathbf{A}\mathcal{P}^T_{\mathbf{C}_1}\|_F^2 &= \|\mathbf{A} - \mathcal{P}_{\mathbf{C}}\mathbf{A} + \mathcal{P}_{\mathbf{C}}\mathbf{A} - \mathcal{P}_{\mathbf{C}}\mathbf{A}\mathcal{P}^T_{\mathbf{C}_1}\|_F^2 \\ &= \|\bar{\mathcal{P}}_{\mathbf{C}}\mathbf{A} + \mathcal{P}_{\mathbf{C}}\mathbf{A}\bar{\mathcal{P}}^T_{\mathbf{C}_1}\|_F^2 \;=\; \|\bar{\mathcal{P}}_{\mathbf{C}}\mathbf{A}\|_F^2 + \|\mathcal{P}_{\mathbf{C}}\mathbf{A}\bar{\mathcal{P}}^T_{\mathbf{C}_1}\|_F^2,\end{aligned}$$

where the last equalities follow from $\mathcal{P}_{\mathbf{C}} \perp \bar{\mathcal{P}}_{\mathbf{C}}$. Since $\mathrm{span}(\mathbf{C}_1) \subset \mathrm{span}(\mathbf{C})$, we have $\|\mathcal{P}_{\mathbf{C}}\mathbf{A}\bar{\mathcal{P}}^T_{\mathbf{C}_1}\|_F^2 \geq \|\mathcal{P}_{\mathbf{C}}\mathbf{A}\bar{\mathcal{P}}^T_{\mathbf{C}}\|_F^2$, which proves the lemma. ■

B.3 The Proof of Theorem 8

The error bound follows directly from Lemma 2 and Corollary 7. The near-optimal column selection algorithm costs $O\big(mk^2\varepsilon^{-4/3} + nk^3\varepsilon^{-2/3}\big) + T_{\text{Multiply}}\big(mnk\varepsilon^{-2/3}\big)$ time to construct $\mathbf{C}$ and $O\big(nk^2\varepsilon^{-4/3} + mk^3\varepsilon^{-2/3}\big) + T_{\text{Multiply}}\big(mnk\varepsilon^{-2/3}\big)$ time to construct $\mathbf{R}_1$. Then the adaptive sampling algorithm costs $O\big(nk^2\varepsilon^{-2}\big) + T_{\text{Multiply}}\big(mnk\varepsilon^{-1}\big)$ time to construct $\mathbf{R}_2$. Computing the Moore-Penrose inverses of $\mathbf{C}$ and $\mathbf{R}$ costs $O(mc^2) + O(nr^2) = O\big(mk^2\varepsilon^{-2} + nk^2\varepsilon^{-4}\big)$ time. The multiplication of $\mathbf{C}^{\dagger}\mathbf{A}\mathbf{R}^{\dagger}$ costs $T_{\text{Multiply}}(mnc) = T_{\text{Multiply}}(mnk\varepsilon^{-1})$ time. So the total time complexity is $O\big((m+n)k^3\varepsilon^{-2/3} + mk^2\varepsilon^{-2} + nk^2\varepsilon^{-4}\big) + T_{\text{Multiply}}\big(mnk\varepsilon^{-1}\big)$.

B.4 The Proof of Theorem 10

The error bound follows immediately from Lemma 2 and Corollary 7. The near-optimal column selection algorithm costs $O\big(mk^2\varepsilon^{-4/3} + mk^3\varepsilon^{-2/3}\big) + T_{\text{Multiply}}\big(m^2k\varepsilon^{-2/3}\big)$ time to select $c_1 = O(k\varepsilon^{-1})$ columns of $\mathbf{A}$ construct $\mathbf{C}_1$. Then the adaptive sampling algorithm costs $O\big(mk^2\varepsilon^{-2}\big) + T_{\text{Multiply}}\big(m^2k\varepsilon^{-1}\big)$ time to select $c_2 = O(k\varepsilon^{-2})$ columns construct $\mathbf{C}_2$. Finally it costs $O(mc^2) + T_{\text{Multiply}}(m^2c) = O(mk^2\varepsilon^{-4}) + T_{\text{Multiply}}\big(m^2k\varepsilon^{-2}\big)$ time to construct the intersection matrix $\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{A}(\mathbf{C}^{\dagger})^T$. So the total time complexity is $O\big(mk^2\varepsilon^{-4} + mk^3\varepsilon^{-2/3}\big) + T_{\text{Multiply}}\big(m^2k\varepsilon^{-2}\big)$.

Appendix C. Proofs of the Lower Error Bounds

In Appendix C.1 we construct two adversarial cases which will be used throughout this appendix. In Appendix C.2 we prove the lower bounds of the standard Nyström method. In Appendix C.3 we prove the lower bounds of the ensemble Nyström method. Theorems 20, 21, 22, 24, and 25 are used for proving Theorem 12.

C.1 Construction of the Adversarial Cases

We now consider the construction of adversarial cases for the spectral norm bounds and the Frobenius norm and nuclear norm bounds, respectively.

C.1.1 THE ADVERSARIAL CASE FOR THE SPECTRAL NORM BOUND

We construct an $m\times m$ positive definite matrix $\mathbf{B}$ as follows:

$$\mathbf{B} = (1-\alpha)\mathbf{I}_m + \alpha\mathbf{1}_m\mathbf{1}_m^T = \begin{bmatrix} 1 & \alpha & \cdots & \alpha \\ \alpha & 1 & \cdots & \alpha \\ \vdots & \vdots & \ddots & \vdots \\ \alpha & \alpha & \cdots & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{W} & \mathbf{B}_{21}^T \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix}, \tag{9}$$

where $\alpha \in [0,1)$. It is easy to verify $\mathbf{x}^T\mathbf{B}\mathbf{x} > 0$ for any nonzero $\mathbf{x} \in \mathbb{R}^m$. We show some properties of $\mathbf{B}$ in Lemma 18.

Lemma 18 *Let $\mathbf{B}_k$ be the best rank-k approximation to the matrix $\mathbf{B}$ defined in (9). Then we have that*

$$\begin{aligned} \|\mathbf{B}\|_F &= \sqrt{m^2\alpha^2 + m(1-\alpha^2)}, & \|\mathbf{B}-\mathbf{B}_k\|_F &= \sqrt{m-k}\,(1-\alpha), \\ \|\mathbf{B}\|_2 &= 1+m\alpha-\alpha\,, & \|\mathbf{B}-\mathbf{B}_k\|_2 &= 1-\alpha, \\ \|\mathbf{B}\|_* &= m, & \|\mathbf{B}-\mathbf{B}_k\|_* &= (m-k)(1-\alpha), \end{aligned}$$

where $1 \le k \le m-1$.

Proof The squared Frobenius norm of $\mathbf{B}$ is

$$\|\mathbf{B}\|_F^2 = \sum_{i,j} b_{ij}^2 = m + (m^2-m)\alpha^2.$$

Then we study the singular values of $\mathbf{B}$. Since $\mathbf{B}$ is SPSD, here we do not distinguish between its singular values and eigenvalues.

The spectral norm, that is, the largest singular value, of $\mathbf{B}$ is

$$\|\mathbf{B}\|_2 = \sigma_1 = \lambda_1 = \max_{\|\mathbf{x}\|_2\le 1} \mathbf{x}^T\mathbf{B}\mathbf{x} = \max_{\|\mathbf{x}\|_2\le 1} (1-\alpha)\|\mathbf{x}\|_2^2 + \alpha(\mathbf{1}_m^T\mathbf{x})^2 = 1-\alpha+m\alpha,$$

where the maximum is attained when $\mathbf{x} = \frac{1}{\sqrt{m}}\mathbf{1}_m$. Thus $\mathbf{u}_1 = \frac{1}{\sqrt{m}}\mathbf{1}_m$ is the top singular vector of $\mathbf{B}$. Then the projection of $\mathbf{B}$ onto the subspace orthogonal to $\mathbf{u}_1$ is

$$\mathbf{B}_{1\perp} \triangleq \mathbf{B} - \mathbf{B}_1 = \mathbf{B} - \sigma_1\mathbf{u}_1\mathbf{u}_1^T = \frac{1-\alpha}{m}(m\mathbf{I}_m - \mathbf{1}_m\mathbf{1}_m^T).$$

Then for all $j > 1$, the j-th top eigenvalue σ_j and eigenvector $\mathbf{u}_j$, that is, the singular value and singular vector, of $\mathbf{B}$ satisfy

$$\sigma_j\mathbf{u}_j = \mathbf{B}\mathbf{u}_j = \mathbf{B}_{1\perp}\mathbf{u}_j = \frac{1-\alpha}{m}\big(m\mathbf{u}_j - (\mathbf{1}_m^T\mathbf{u}_j)\mathbf{1}_m\big) = \frac{1-\alpha}{m}(m\mathbf{u}_j - \mathbf{0}),$$

where the last equality follows from $\mathbf{u}_j \perp \mathbf{u}_1$, that is, $\mathbf{1}_m^T \mathbf{u}_j = 0$. Thus $\sigma_j = 1-\alpha$, and

$$\|\mathbf{B}-\mathbf{B}_k\|_2 \;=\; \sigma_{k+1} \;=\; 1-\alpha$$

for all $1 \le k < m$. Finally we have that

$$\begin{aligned}
\|\mathbf{B}-\mathbf{B}_k\|_F^2 &= \|\mathbf{B}\|_F^2 - \sum_{i=1}^{k}\sigma_i^2 \;=\; (m-k)(1-\alpha)^2,\\
\|\mathbf{B}-\mathbf{B}_k\|_* &= (m-k)\sigma_2 \;=\; (m-k)(1-\alpha),\\
\|\mathbf{B}\|_* &= \sum_{i=1}^{m}\sigma_i \;=\; (1+m\alpha-\alpha)+(m-1)(1-\alpha) \;=\; m,
\end{aligned}$$

which complete our proofs. ■

C.1.2 THE ADVERSARIAL CASE FOR THE FROBENIUS NORM AND NUCLEAR NORM BOUNDS

Then we construct another adversarial case for proving the Frobenius norm and nuclear norm bounds. Let $\mathbf{B}$ be a $p \times p$ matrix with diagonal entries equal to one and off-diagonal entries equal to α. Let $m = kp$ and we construct an $m \times m$ block diagonal matrix $\mathbf{A}$ as follows:

$$\mathbf{A} \;=\; \mathsf{BlkDiag}(\underbrace{\mathbf{B},\cdots,\mathbf{B}}_{k \text{ blocks}}) \;=\; \begin{bmatrix} \mathbf{B} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{B} \end{bmatrix}. \tag{10}$$

Lemma 19 *Let $\mathbf{A}_k$ be the best rank-k approximation to the matrix $\mathbf{A}$ defined in (10). Then we have that*

$$\begin{aligned}
\sigma_1(\mathbf{A}) &= \cdots = \sigma_k(\mathbf{A}) = 1+p\alpha-\alpha,\\
\sigma_{k+1}(\mathbf{A}) &= \cdots = \sigma_m(\mathbf{A}) = 1-\alpha,\\
\left\|\mathbf{A}-\mathbf{A}_k\right\|_F &= (1-\alpha)\sqrt{m-k},\\
\left\|\mathbf{A}-\mathbf{A}_k\right\|_* &= (1-\alpha)(m-k).
\end{aligned}$$

Lemma 19 can be easily proved using Lemma 18.

C.2 Lower Bounds of the Standard Nyström Method

Theorem 20 *For an $m \times m$ matrix $\mathbf{B}$ with diagonal entries equal to one and off-diagonal entries equal to $\alpha \in [0,1)$, the approximation error incurred by the standard Nyström method is lower bounded by*

$$\begin{aligned}
\left\|\mathbf{B}-\tilde{\mathbf{B}}_c^{nys}\right\|_F &\ge (1-\alpha)\sqrt{(m-c)\Big(1+\frac{m+c+\frac{2}{\alpha}-2}{(c+\frac{1-\alpha}{\alpha})^2}\Big)},\\
\left\|\mathbf{B}-\tilde{\mathbf{B}}_c^{nys}\right\|_2 &\ge \frac{(1-\alpha)\Big(m+\frac{1-\alpha}{\alpha}\Big)}{c+\frac{1-\alpha}{\alpha}},\\
\left\|\mathbf{B}-\tilde{\mathbf{B}}_c^{nys}\right\|_* &\ge (m-c)(1-\alpha)\frac{1+c\alpha}{1+c\alpha-\alpha}.
\end{aligned}$$

Furthermore, the matrix $(\mathbf{B}-\tilde{\mathbf{B}}_c^{nys})$ *is SPSD.*

Proof The matrix $\mathbf{B}$ is partitioned as in (9). The residual of the Nyström approximation is

$$\|\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\|_\xi \;=\; \|\mathbf{B}_{22}-\mathbf{B}_{21}\mathbf{W}^\dagger\mathbf{B}_{21}^T\|_\xi, \tag{11}$$

where $\xi = 2$, F, or $*$. Since $\mathbf{W} = (1-\alpha)\mathbf{I}_c + \alpha\mathbf{1}_c\mathbf{1}_c^T$ is nonsingular when $\alpha \in [0,1)$, so $\mathbf{W}^\dagger = \mathbf{W}^{-1}$. We apply the Sherman-Morrison-Woodbury formula

$$(\mathbf{A}+\mathbf{BCD})^{-1} \;=\; \mathbf{A}^{-1}-\mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B})^{-1}\mathbf{DA}^{-1}$$

to compute $\mathbf{W}^\dagger$, yielding

$$\mathbf{W}^\dagger \;=\; \frac{1}{1-\alpha}\mathbf{I}_c - \frac{\alpha}{(1-\alpha)(1-\alpha+c\alpha)}\mathbf{1}_c\mathbf{1}_c^T.$$

According to the construction, $\mathbf{B}_{21}$ is an $(m-c)\times c$ matrix with all entries equal to α, it follows that $\mathbf{B}_{21}\mathbf{W}^\dagger\mathbf{B}_{21}^T$ is an $(m-c)\times(m-c)$ matrix with all entries equal to

$$\eta \;\triangleq\; \alpha^2\mathbf{1}_c^T\mathbf{W}^\dagger\mathbf{1}_c \;=\; \frac{c\alpha^2}{1-\alpha+c\alpha}. \tag{12}$$

Then we obtain that

$$\mathbf{B}_{22}-\mathbf{B}_{21}\mathbf{W}^\dagger\mathbf{B}_{21}^T \;\;=\;\; (1-\alpha)\mathbf{I}_{m-c}+(\alpha-\eta)\mathbf{1}_{m-c}\mathbf{1}_{m-c}^T. \tag{13}$$

It is easy to check that $\eta \le \alpha \le 1$, thus the matrix $(1-\alpha)\mathbf{I}_{m-c}+(\alpha-\eta)\mathbf{1}_{m-c}\mathbf{1}_{m-c}^T$ is SPSD, and so is $(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}})$.

Combining (11) and (13), we have that

$$\begin{aligned}
\|\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\|_F^2 &= \big\|(1-\alpha)\mathbf{I}_{m-c}+(\alpha-\eta)\mathbf{1}_{m-c}\mathbf{1}_{m-c}^T\big\|_F^2 \\
&= (m-c)\big(1-\eta\big)^2+\Big((m-c)^2-(m-c)\Big)\big(\alpha-\eta\big)^2 \\
&= (m-c)(1-\alpha)^2\Big(1+\frac{\alpha^2(m+c)+2(\alpha-\alpha^2)}{(1-\alpha+c\alpha)^2}\Big) \\
&= (m-c)(1-\alpha)^2\Big(1+\frac{m+c+\frac{2}{\alpha}-2}{(c+\frac{1-\alpha}{\alpha})^2}\Big),
\end{aligned} \tag{14}$$

which proves the Frobenius norm of the residual.

Now we compute the spectral norm of the residual. Based on the results above we have that

$$\big\|\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big\|_2 \;=\; \big\|(1-\alpha)\mathbf{I}_{m-c}+(\alpha-\eta)\mathbf{1}_{m-c}\mathbf{1}_{m-c}^T\big\|_2.$$

Similar to the proof of Lemma 18, it is easily obtained that $\frac{1}{\sqrt{m-c}}\mathbf{1}_{m-c}$ is the top singular vector of the SPSD matrix $(1-\alpha)\mathbf{I}_{m-c}+(\alpha-\eta)\mathbf{1}_{m-c}\mathbf{1}_{m-c}^T$, so the top singular value is

$$\sigma_1\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big) \;=\; (m-c)(\alpha-\eta)+1-\alpha \;=\; \frac{(1-\alpha)\Big(m+\frac{1-\alpha}{\alpha}\Big)}{c+\frac{1-\alpha}{\alpha}}, \tag{15}$$

which proves the spectral norm bound because $\|\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\|_2 = \sigma_1\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big)$.

It is also easy to show the rest singular values obey

$$\begin{aligned}\sigma_2\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big) &= \cdots = \sigma_{m-c}\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big) \geq 0,\\ \sigma_{m-c+1}\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big) &= \cdots = \sigma_m\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big) = 0.\end{aligned}$$

Thus we have, for $i = 2,\cdots,m-c$,

$$\sigma_i^2\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big) = \frac{\|\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\|_F^2 - \sigma_1^2\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big)}{m-c-1} = (1-\alpha)^2.$$

The nuclear norm of the residual $\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big)$ is

$$\begin{aligned}\|\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\|_* &= \sum_{i=1}^{m}\sigma\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big)\\ &= \sigma_1\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big) + (m-c-1)\,\sigma_2\big(\mathbf{B}-\tilde{\mathbf{B}}_c^{\text{nys}}\big)\\ &= (m-c)(1-\eta)\\ &= (m-c)(1-\alpha)\Big(1+\frac{1}{c+\frac{1-\alpha}{\alpha}}\Big). \qquad (16)\end{aligned}$$

The theorem follows from equalities (14), (15), and (16). ■

Now we use the matrix $\mathbf{A}$ constructed in (10) to show the Frobenius norm and nuclear norm lower bound. The bound is stronger than the one in Theorem 20 by a factor of k.

Theorem 21 *For the $m\times m$ SPSD matrix $\mathbf{A}$ defined in (10), the approximation error incurred by the standard Nyström method is lower bounded by*

$$\begin{aligned}\big\|\mathbf{A}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\big\|_F &\geq (1-\alpha)\sqrt{m-c-k+\frac{k(m+\frac{1-\alpha}{\alpha}k)^2}{(c+\frac{1-\alpha}{\alpha}k)^2}},\\ \big\|\mathbf{A}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\big\|_* &\geq (1-\alpha)(m-c)\Big(1+\frac{k}{c+\frac{1-\alpha}{\alpha}k}\Big),\end{aligned}$$

where $k<m$ is an arbitrary positive integer.

Proof Let $\mathbf{C}$ consist of c column sampled from $\mathbf{A}$ and $\hat{\mathbf{C}}_i$ consist of c_i columns sampled from the i-th block diagonal matrix in $\mathbf{A}$. Without loss of generality, we assume $\hat{\mathbf{C}}_i$ consists of the first c_i columns of $\mathbf{B}$, and accordingly $\hat{\mathbf{W}}_i$ consists of the top left $c_i\times c_i$ block of $\mathbf{B}$. Thus $\mathbf{C} = \mathsf{BlkDiag}\big(\hat{\mathbf{C}}_1,\cdots,\hat{\mathbf{C}}_k\big)$

and $\mathbf{W} = \mathsf{BlkDiag}\big(\hat{\mathbf{W}}_1, \cdots, \hat{\mathbf{W}}_k\big)$.

$$\begin{aligned}
\tilde{\mathbf{A}}_c^{\text{nys}} \;=\; \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C} \;&=\; \begin{bmatrix} \hat{\mathbf{C}}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{\mathbf{C}}_k \end{bmatrix} \begin{bmatrix} \hat{\mathbf{W}}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{\mathbf{W}}_k \end{bmatrix}^{\dagger} \begin{bmatrix} \hat{\mathbf{C}}_1^T & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{\mathbf{C}}_k^T \end{bmatrix} \\
&= \begin{bmatrix} \hat{\mathbf{C}}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{\mathbf{C}}_k \end{bmatrix} \begin{bmatrix} \hat{\mathbf{W}}_1^{\dagger} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{\mathbf{W}}_k^{\dagger} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{C}}_1^T & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{\mathbf{C}}_k^T \end{bmatrix} \\
&= \begin{bmatrix} \hat{\mathbf{C}}_1\hat{\mathbf{W}}_1^{\dagger}\hat{\mathbf{C}}_1^T & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \hat{\mathbf{C}}_k\hat{\mathbf{W}}_k^{\dagger}\hat{\mathbf{C}}_k^T \end{bmatrix}. \qquad (17)
\end{aligned}$$

Then it follows from Theorem 20 that

$$\begin{aligned}
\big\|\mathbf{A} - \tilde{\mathbf{A}}_c^{\text{nys}}\big\|_F^2 \;&=\; \sum_{i=1}^{k} \big\|\mathbf{B} - \hat{\mathbf{C}}_i\hat{\mathbf{W}}_i^{\dagger}\hat{\mathbf{C}}_i^T\big\|_F^2 \\
&=\; \sum_{i=1}^{k} (p - c_i)(1-\alpha)^2\Big(1 + \frac{p + c_i + 2\frac{1-\alpha}{\alpha}}{(c_i + \frac{1-\alpha}{\alpha})^2}\Big) \\
&=\; (1-\alpha)^2 \sum_{i=1}^{k} (\hat{p} - \hat{c}_i)\Big(1 + \frac{\hat{p} + \hat{c}_i}{\hat{c}_i^2}\Big) \\
&=\; (1-\alpha)^2\Big(m - c - k + \hat{p}^2 \sum_{i=1}^{k} \hat{c}_i^{-2}\Big),
\end{aligned}$$

where $\hat{p} = p + \frac{1-\alpha}{\alpha}$ and $\hat{c}_i = c_i + \frac{1-\alpha}{\alpha}$. Since $\sum_{i=1}^{k} \hat{c}_i = c + \frac{1-\alpha}{\alpha}k \triangleq \hat{c}$, the term $\sum_{i=1}^{k} \hat{c}_i^{-2}$ is minimized when $\hat{c}_1 = \cdots = \hat{c}_k$. Thus $\sum_{i=1}^{k} \hat{c}_i^{-2} = k\frac{k^2}{\hat{c}^2} = k^3\hat{c}^{-2}$. Finally we have that

$$\begin{aligned}
\big\|\mathbf{A} - \tilde{\mathbf{A}}_c^{\text{nys}}\big\|_F^2 \;&=\; (1-\alpha)^2\Big(m - c - k + \hat{p}^2 \sum_{i=1}^{k} \hat{c}_i^{-2}\Big) \\
&\geq\; (1-\alpha)^2\Big(m - c - k + \frac{k(m + \frac{1-\alpha}{\alpha}k)^2}{(c + \frac{1-\alpha}{\alpha}k)^2}\Big),
\end{aligned}$$

by which the Frobenius norm bound follows.

Since the matrices $\mathbf{B} - \hat{\mathbf{C}}_i\hat{\mathbf{W}}_i^{\dagger}\hat{\mathbf{C}}_i^T$ are all SPSD by Theorem 20, so the matrix $(\mathbf{A} - \tilde{\mathbf{A}}_c^{\text{nys}})$ is also SPSD. We have that

$$\begin{aligned}
\big\|\mathbf{A} - \tilde{\mathbf{A}}_c^{\text{nys}}\big\|_* \;&=\; \sum_{i=1}^{k} \big\|\mathbf{B} - \hat{\mathbf{C}}_i\hat{\mathbf{W}}_i^{\dagger}\hat{\mathbf{C}}_i^T\big\|_* \\
&\geq\; (1-\alpha) \sum_{i=1}^{k} (p - c_i)\Big(1 + \frac{1}{c_i + \frac{1-\alpha}{\alpha}}\Big) \\
&\geq\; (1-\alpha)\, k\,(\frac{m}{k} - \frac{c}{k})\Big(1 + \frac{1}{c/k + \frac{1-\alpha}{\alpha}}\Big) \\
&=\; (1-\alpha)(m-c)\Big(1 + \frac{k}{c + \frac{1-\alpha}{\alpha}k}\Big),
\end{aligned}$$

where the former inequality follows from Theorem 20, and the latter inequality follows by minimizing w.r.t. $c_1, \cdots, c_k$ subjecting to $c_1 + \cdots + c_k = c$. ■

Theorem 22 *There exists an $m \times m$ SPSD matrix $\mathbf{A}$ such that the approximation error incurred by the standard Nyström method is lower bounded by*

$$\begin{aligned}
\frac{\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\right\|_F}{\left\|\mathbf{A} - \mathbf{A}_k\right\|_F} &\geq \sqrt{1 + \frac{m^2k - c^3}{c^2(m-k)}}, \\
\frac{\|\mathbf{A} - \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\|_2}{\|\mathbf{A} - \mathbf{A}_k\|_2} &\geq \frac{m}{c}, \\
\frac{\|\mathbf{A} - \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\|_*}{\|\mathbf{A} - \mathbf{A}_k\|_*} &\geq \frac{m-c}{m-k}\Big(1 + \frac{k}{c}\Big),
\end{aligned}$$

where $k < m$ is an arbitrary positive integer.

Proof For the spectral norm bound we use the matrix $\mathbf{A}$ constructed in (9) and set $\alpha \to 1$, then it follows directly from Lemma 18 and Theorem 20. For the Frobenius norm and nuclear norm bounds, we use the matrix $\mathbf{A}$ constructed in (10) and set $\alpha \to 1$, then it follows directly from Lemma 19 and Theorem 21. ■

C.3 Lower Bounds of the Ensemble Nyström Method

The ensemble Nyström method (Kumar et al., 2009) is previously defined in (2). To derive lower bounds of the ensemble Nyström method, we assume that the t samples are non-overlapping. According to the construction of the matrix $\mathbf{B}$ in (9), each of the t non-overlapping samples are equally "important", so without loss of generality we set the t samples with equal weights: $\mu^{(1)} = \cdots = \mu^{(t)} = \frac{1}{t}$.

Lemma 23 *Assume that the ensemble Nyström method selects a collection of t samples, each sample $\mathbf{C}^{(i)}$ $(i = 1, \cdots, t)$ contains c columns of $\mathbf{B}$ without overlapping. For an $m \times m$ matrix $\mathbf{B}$ with all diagonal entries equal to one and off-diagonal entries equal to $\alpha \in [0, 1)$, the approximation error incurred by the ensemble Nyström method is lower bounded by*

$$\begin{aligned}
\left\|\mathbf{B} - \tilde{\mathbf{B}}_{t,c}^{ens}\right\|_F &\geq (1-\alpha)\sqrt{\Big(m - 2c + \frac{c}{t}\Big)\Big(1 + \frac{m + \frac{c}{t} + \frac{2}{\alpha} - 2}{(c + \frac{1-\alpha}{\alpha})^2}\Big)}, \\
\left\|\mathbf{B} - \tilde{\mathbf{B}}_{t,c}^{ens}\right\|_* &\geq (1-\alpha)(m-c)\frac{c + \frac{1}{\alpha}}{c + \frac{1-\alpha}{\alpha}}.
\end{aligned}$$

where $\tilde{\mathbf{B}}_{t,c}^{ens} = \frac{1}{t}\sum_{i=1}^{t} \mathbf{C}^{(i)}{\mathbf{W}^{(i)}}^{\dagger}{\mathbf{C}^{(i)}}^T$. Furthermore, the matrix $(\mathbf{B} - \tilde{\mathbf{B}}_{t,c}^{ens})$ is SPSD.

Proof We use the matrix $\mathbf{B}$ constructed in (9). It is easy to check that $\mathbf{W}^{(1)} = \cdots = \mathbf{W}^{(t)}$, so we use the notation $\mathbf{W}$ instead. We assume that the samples contain the firs tc columns of $\mathbf{B}$ and each sample contains neighboring columns, that is,

$$\mathbf{B} = \left[\mathbf{C}^{(1)}, \cdots, \mathbf{C}^{(t)}, \mathbf{B}_{(tc+1):m}\right].$$

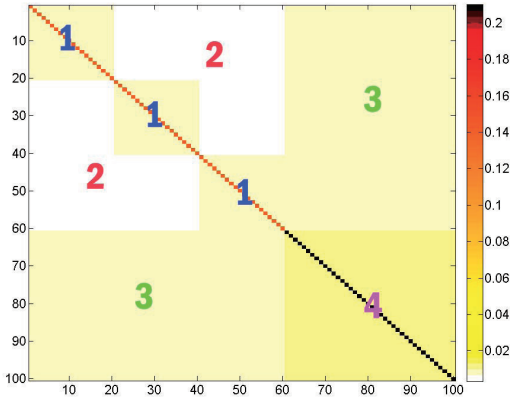


Figure 8: An illustration of the matrix $\mathbf{B}-\mathbf{B}_{t,c}^{\text{ens}}$ for the ensemble Nyström method where $\mathbf{B}$ is defined in (9). Here we set $m=100$, $c=20$, $\alpha=0.8$, and $t=3$. For the ensemble Nyström method without overlapping, the matrix $\mathbf{B}-\mathbf{B}_{t,c}^{\text{ens}}$ can always be partitioned into four regions as annotated.

If a sample $\mathbf{C}$ contains the first c columns of $\mathbf{B}$, then

$$\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T} = \begin{bmatrix} \mathbf{W} & \mathbf{B}_{21}^{T} \\ \mathbf{B}_{21} & \mathbf{B}_{21}\mathbf{W}^{\dagger}\mathbf{B}_{21}^{T} \end{bmatrix} \quad \text{and} \quad \mathbf{B}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{22}-\mathbf{B}_{21}\mathbf{W}^{\dagger}\mathbf{B}_{21}^{T} \end{bmatrix};$$

otherwise, after permuting the rows and columns of $\mathbf{B}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}$, we get the same result:

$$\Pi\big(\mathbf{B}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\big)\Pi^{T} = \mathbf{B}-\Pi\big(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\big)\Pi^{T} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{22}-\mathbf{B}_{21}\mathbf{W}^{\dagger}\mathbf{B}_{21}^{T} \end{bmatrix},$$

where Π is a permutation matrix. As was shown in Equation (12), $\mathbf{B}_{21}\mathbf{W}^{\dagger}\mathbf{B}_{21}^{T}$ is an $(m-c)\times(m-c)$ matrix with all entries equal to

$$\eta = \frac{c\alpha^{2}}{1-\alpha+c\alpha}.$$

Based on the properties of the matrix $\mathbf{B}-\mathbf{C}^{(i)}{\mathbf{W}^{(i)}}^{\dagger}{\mathbf{C}^{(i)}}^{T}$, we study the values of the entries of $\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}}$. We can express it as

$$\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}} = \mathbf{B}-\frac{1}{t}\sum_{i=1}^{t}\mathbf{C}^{(i)}{\mathbf{W}^{(i)}}^{\dagger}{\mathbf{C}^{(i)}}^{T} = \frac{1}{t}\sum_{i=1}^{t}\Big(\mathbf{B}-\mathbf{C}^{(i)}\mathbf{W}^{\dagger}{\mathbf{C}^{(i)}}^{T}\Big), \tag{18}$$

and then a discreet examination reveals that $\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}}$ can be partitioned into four kinds of regions as illustrated in Figure 8. We annotate the regions in the figure and summarize the values of entries in each region in the table below. (Region 1 and 4 are further partitioned into diagonal entries and off-diagonal entries.)

Region	1 (diag)	1 (off-diag)	2	3	4 (diag)	4 (off-diag)
#Entries	tc	tc^2-tc	$(tc)^2-tc^2$	$2tc(m-tc)$	$m-tc$	$(m-tc)^2-(m-tc)$
Value	$\frac{t-1}{t}(1-\eta)$	$\frac{t-1}{t}(\alpha-\eta)$	$\frac{t-2}{t}(\alpha-\eta)$	$\frac{t-1}{t}(\alpha-\eta)$	$1-\eta$	$\alpha-\eta$

Now we do summation over the entries of $\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}}$ to compute its squared Frobenius norm:

$$\begin{aligned}
\big\|\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}}\big\|_F^2 &= tc\Big[\frac{t-1}{t}(1-\eta)\Big]^2+\cdots+\big[(m-tc)^2-(m-tc)\big](\alpha-\eta)^2 \\
&= (1-\alpha)(1+\alpha-2\eta)(m-2c+\frac{c}{t})+(\alpha-\eta)^2\Big(4c^2-4cm+m^2+\frac{2cm-3c^2}{t}\Big) \\
&= (1-\alpha)^2\Big(m-2c+\frac{c}{t}\Big)+\frac{(1-\alpha)^2}{(c+\frac{1-\alpha}{\alpha})^2}\Big[(m-2c+\frac{c}{t})(\frac{2}{\alpha}-2+m)+\frac{c(m-c)}{t}\Big] \\
&\geq (1-\alpha)^2\Big(m-2c+\frac{c}{t}\Big)\Big(1+\frac{m+\frac{c}{t}+\frac{2}{\alpha}-2}{(c+\frac{1-\alpha}{\alpha})^2}\Big),
\end{aligned}$$

where the last inequality follows from $\frac{c(m-c)}{t}=\frac{c}{t}\Big((m-2c+\frac{c}{t})+(c-\frac{c}{t})\Big)\geq\frac{c}{t}\Big(m-2c+\frac{c}{t}\Big)$.

Furthermore, since the matrices $\mathbf{B}-\mathbf{C}^{(i)}\mathbf{W}^{\dagger}{\mathbf{C}^{(i)}}^T$ are all SPSD by Theorem 20, so their sum is also SPSD. Then the SPSD property of $(\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}})$ follows from (18). Therefore, the nuclear norm of $(\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}})$ equals to the matrix trace, that is,

$$\begin{aligned}
\big\|\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}}\big\|_* &= \text{tr}\big(\mathbf{B}-\tilde{\mathbf{B}}_{t,c}^{\text{ens}}\big) \\
&= tc\cdot\frac{t-1}{t}(1-\eta)+(m-tc)\cdot(1-\eta) \\
&= (1-\alpha)(m-c)\frac{c+\frac{1}{\alpha}}{c+\frac{1-\alpha}{\alpha}},
\end{aligned}$$

which proves the nuclear norm bound in the lemma. ■

Theorem 24 *Assume that the ensemble Nyström method selects a collection of t samples, each sample $\mathbf{C}^{(i)}$ $(i=1,\cdots,t)$ contains c columns of $\mathbf{A}$ without overlapping. For a the matrix $\mathbf{A}$ defined in (10), the approximation error incurred by the ensemble Nyström method is lower bounded by*

$$\begin{aligned}
\big\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{ens}\big\|_F &\geq (1-\alpha)\sqrt{\Big(m-2c+\frac{c}{t}-k\Big)+k\Big(\frac{m-c+\frac{c}{t}+k\frac{1-\alpha}{\alpha}}{c+k\frac{1-\alpha}{\alpha}}\Big)^2}, \\
\big\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{ens}\big\|_* &\geq (1-\alpha)(m-c)\frac{c+\frac{1}{\alpha}k}{c+\frac{1-\alpha}{\alpha}k},
\end{aligned}$$

where $\tilde{\mathbf{A}}_{t,c}^{ens}=\frac{1}{t}\sum_{i=1}^{t}\mathbf{C}^{(i)}{\mathbf{W}^{(i)}}^{\dagger}{\mathbf{C}^{(i)}}^T$.

Proof According to the construction of $\mathbf{A}$ in (10), the i-th sample $\mathbf{C}^{(i)}$ is also block diagonal. We denote it by $\mathbf{C}^{(i)}=\mathsf{BlkDiag}\big(\hat{\mathbf{C}}_1^{(i)},\cdots,\hat{\mathbf{C}}_k^{(i)}\big)$. Akin to (17), we have

$$\tilde{\mathbf{A}}_{t,c}^{\text{ens}} = \begin{bmatrix} \frac{1}{t}\sum_{i=1}^{t}\hat{\mathbf{C}}_1^{(i)}\hat{\mathbf{W}}_1^{\dagger}\big(\hat{\mathbf{C}}_1^{(i)}\big)^T & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \frac{1}{t}\sum_{i=1}^{t}\hat{\mathbf{C}}_k^{(i)}\hat{\mathbf{W}}_k^{\dagger}\big(\hat{\mathbf{C}}_k^{(i)}\big)^T \end{bmatrix}.$$

Thus the approximation error of the ensemble Nyström method is

$$\begin{aligned}\left\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{\text{ens}}\right\|_F^2 &= \sum_{j=1}^{k}\left\|\mathbf{B}-\frac{1}{t}\sum_{i=1}^{t}\hat{\mathbf{C}}_j^{(i)}\hat{\mathbf{W}}_j^{\dagger}(\hat{\mathbf{C}}_j^{(i)})^T\right\|_F^2 \\ &\geq (1-\alpha)^2\sum_{j=1}^{k}\Big(p-2c_j+\frac{c_j}{t}\Big)\Big(1+\frac{p+\frac{c_j}{t}+\frac{2}{\alpha}-2}{(c_j+\frac{1-\alpha}{\alpha})^2}\Big) \\ &= (1-\alpha)^2\Bigg[\Big(m-2c+\frac{c}{t}\Big)+\sum_{j=1}^{k}\Big(p-2c_j+\frac{c_j}{t}\Big)\frac{p+\frac{c_j}{t}+\frac{2(1-\alpha)}{\alpha}}{(c_j+\frac{1-\alpha}{\alpha})^2}\Bigg],\end{aligned}$$

where the inequality follows from Lemma 23, and the last equality follows from $\sum_{j=1}^{k}c_j=c$ and $kp=m$. The summation in the last equality equals to

$$\begin{aligned}&\sum_{j=1}^{k}\Bigg[\Big(p+\frac{c_j}{t}+\frac{2(1-\alpha)}{\alpha}\Big)-2\Big(c_j+\frac{1-\alpha}{\alpha}\Big)\Bigg]\frac{p+\frac{c_j}{t}+\frac{2(1-\alpha)}{\alpha}}{(c_j+\frac{1-\alpha}{\alpha})^2} \\ &= -k+\sum_{j=1}^{k}\left(\frac{p+\frac{c_j}{t}+\frac{2(1-\alpha)}{\alpha}}{c_j+\frac{1-\alpha}{\alpha}}-1\right)^2 \\ &\geq -k+k\left(\frac{m-c+\frac{c}{t}+k\frac{1-\alpha}{\alpha}}{c+k\frac{1-\alpha}{\alpha}}\right)^2.\end{aligned}$$

Here the inequality holds because the function is minimized when $c_1=\cdots=c_k=c/k$. Finally we have that

$$\left\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{\text{ens}}\right\|_F^2 \geq (1-\alpha)^2\Bigg[\Big(m-2c+\frac{c}{t}-k\Big)+k\left(\frac{m-c+\frac{c}{t}+k\frac{1-\alpha}{\alpha}}{c+k\frac{1-\alpha}{\alpha}}\right)^2\Bigg],$$

which proves the Frobenius norm bound in the theorem.

Furthermore, since the matrix $\mathbf{B}-\frac{1}{t}\sum_{i=1}^{t}\hat{\mathbf{C}}_j^{(i)}\hat{\mathbf{W}}_j^{\dagger}(\hat{\mathbf{C}}_j^{(i)})^T$ is SPSD by Lemma 23, so the block diagonal matrix $(\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{\text{ens}})$ is also SPSD. Thus we have

$$\left\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{\text{ens}}\right\|_* = (1-\alpha)\sum_{i=1}(p-c_i)\frac{c_i+\frac{1}{\alpha}}{c_i+\frac{1-\alpha}{\alpha}} \geq (1-\alpha)(m-c)\Big(1+\frac{k}{c+\frac{1-\alpha}{\alpha}k}\Big),$$

which proves the nuclear norm bound in the theorem. ■

Theorem 25 *Assume that the ensemble Nyström method selects a collection of t samples, each sample $\mathbf{C}^{(i)}$ $(i=1,\cdots,t)$ contains c columns of $\mathbf{A}$ without overlapping. Then there exists an $m\times m$ SPSD matrix $\mathbf{A}$ such that the relative-error ratio of the ensemble Nyström method is lower bounded by*

$$\begin{aligned}\frac{\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{ens}\|_F}{\|\mathbf{A}-\mathbf{A}_k\|_F} &\geq \sqrt{\frac{m-2c+c/t-k}{m-k}\Big(1+\frac{k(m-2c+c/t)}{c^2}\Big)}, \\ \frac{\|\mathbf{A}-\tilde{\mathbf{A}}_{t,c}^{ens}\|_*}{\|\mathbf{A}-\mathbf{A}_k\|_*} &\geq \frac{m-c}{m-k}\Big(1+\frac{k}{c}\Big),\end{aligned}$$

where $\tilde{\mathbf{A}}_{t,c}^{ens}=\frac{1}{t}\sum_{i=1}^{t}\mathbf{C}^{(i)}{\mathbf{W}^{(i)}}^{\dagger}{\mathbf{C}^{(i)}}^T$.

Proof The theorem follows directly from Theorem 24 and Lemma 19 by setting $\alpha \to 1$. ■

References

A. Ben-Israel and T. N. E. Greville. *Generalized Inverses: Theory and Applications. Second Edition*. Springer, 2003.

M. W. Berry, S. A. Pulatova, and G. W. Stewart. Algorithm 844: computing sparse reduced-rank approximations to sparse matrices. *ACM Transactions on Mathematical Software*, 31(2):252–269, 2005.

J. Bien, Y. Xu, and M. W. Mahoney. CUR from a sparse optimization viewpoint. In *Advances in Neural Information Processing Systems (NIPS)*. 2010.

C. H. Bischof and P. C. Hansen. Structure-preserving and rank-revealing QR-factorizations. *SIAM Journal on Scientific and Statistical Computing*, 12(6):1332–1350, 1991.

C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near-optimal column-based matrix reconstruction. *CoRR*, abs/1103.0995, 2011.

T. F. Chan. Rank revealing QR factorizations. *Linear Algebra and Its Applications*, 88:67–82, 1987.

S. Chandrasekaran and I. C. F. Ipsen. On rank-revealing factorisations. *SIAM Journal on Matrix Analysis and Applications*, 15(2):592–622, 1994.

P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of The American Society for Information Science*, 41(6):391–407, 1990.

A. Deshpande and L. Rademacher. Efficient volume sampling for row/column subset selection. In *Proceedings of the 51st IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 329–338, 2010.

A. Deshpande, L. Rademacher, S. Vempala, and G. Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(2006):225–247, 2006.

P. Drineas and R. Kannan. Pass-efficient algorithms for approximating large matrices. In *Proceeding of the 14th Annual ACM-SIAM Symposium on Dicrete Algorithms (SODA)*, pages 223–232, 2003.

P. Drineas and M. W. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices III: computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36(1): 184–206, 2006.

P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, September 2008.

P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff. Fast approximation of matrix coherence and statistical leverage. In *International Conference on Machine Learning (ICML)*, 2012.

L. V. Foster. Rank and null space calculations using matrix decomposition without column interchanges. *Linear Algebra and its Applications*, 74:47–71, 1986.

C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

A. Frieze, R. Kannan, and S. Vempala. Fast Monte Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, November 2004. ISSN 0004-5411.

A. Gittens and M. W. Mahoney. Revisiting the Nyström method for improved large-scale machine learning. *arXiv preprint arXiv:1303.1849*, 2013.

S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra and Its Applications*, 261:1–21, 1997a.

S. A. Goreinov, N. L. Zamarashkin, and E. E. Tyrtyshnikov. Pseudo-skeleton approximations by matrices of maximal volume. *Mathematical Notes*, 62(4):619–623, 1997b.

M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996.

V. Guruswami and A. K. Sinop. Optimal column-based low-rank matrix reconstruction. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2012.

I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror. Result analysis of the NIPS 2003 feature selection challenge. *Advances in Neural Information Processing Systems (NIPS)*, 2004.

N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

Y. P. Hong and C. T. Pan. Rank-revealing QR factorizations and the singular value decomposition. *Mathematics of Computation*, 58(197):213–232, 1992.

R. Jin, T. Yang, and M. Mahdavi. Improved bound for the Nyström method and its application to kernel classification. *CoRR*, abs/1111.2262, 2011.

S. Kumar, M. Mohri, and A. Talwalkar. Ensemble Nyström method. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling methods for the Nyström method. *Journal of Machine Learning Research*, 13:981–1006, 2012.

F. G. Kuruvilla, P. J. Park, and S. L. Schreiber. Vector algebra in the analysis of genome-wide expression data. *Genome Biology*, 3:research0011–research0011.1, 2002.

M. Li, J. T. Kwok, and B.-L. Lu. Making large-scale Nyström approximation possible. In *International Conference on Machine Learning (ICML)*, 2010.

L. Mackey, A. Talwalkar, and M. I. Jordan. Divide-and-conquer matrix factorization. In *Advances in Neural Information Processing Systems (NIPS)*. 2011.

M. W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

M. W. Mahoney, M. Maggioni, and P. Drineas. Tensor-CUR decompositions for tensor-based data. *SIAM Journal on Matrix Analysis and Applications*, 30(3):957–987, 2008.

C. Mesterharm and M. J. Pazzani. Active learning using on-line algorithms. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2011.

D. Michie, D. J. Spiegelhalter, and C. C. Taylor. Machine learning, neural and statistical classification. 1994.

L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A*, 4(3):519–524, Mar 1987.

G. W. Stewart. Four algorithms for the the efficient computation of truncated pivoted QR approximations to a sparse matrix. *Numerische Mathematik*, 83(2):313–323, 1999.

A. Talwalkar and A. Rostamizadeh. Matrix coherence and the Nyström method. *arXiv preprint arXiv:1004.2008*, 2010.

A. Talwalkar, S. Kumar, and H. Rowley. Large-scale manifold learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1): 71–86, 1991.

E. E. Tyrtyshnikov. Incomplete cross approximation in the mosaic-skeleton method. *Computing*, 64:367–380, 2000.

C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

K. Zhang and J. T. Kwok. Clustered Nyström method for large scale manifold learning and dimension reduction. *IEEE Transactions on Neural Networks*, 21(10):1576–1587, 2010.

K. Zhang, I. W. Tsang, and J. T. Kwok. Improved Nyström low-rank approximation and error analysis. In *International Conference on Machine Learning (ICML)*, 2008.

Journal of Machine Learning Research 14 (2013) 2771-2797 Submitted 1/13; Revised 5/13; Published 9/13

Training Energy-Based Models for Time-Series Imputation

Philémon Brakel PHILEMON.BRAKEL@UGENT.BE
Dirk Stroobandt DIRK.STROOBANDT@UGENT.BE
Benjamin Schrauwen BENJAMIN.SCHRAUWEN@UGENT.BE
Department of Electronics and Information Systems
University of Ghent
Sint-Pietersnieuwstraat 41
9000 Gent, Belgium

Editor: Yoshua Bengio

Abstract

Imputing missing values in high dimensional time-series is a difficult problem. This paper presents a strategy for training energy-based graphical models for imputation directly, bypassing difficulties probabilistic approaches would face. The training strategy is inspired by recent work on optimization-based learning (Domke, 2012) and allows complex neural models with convolutional and recurrent structures to be trained for imputation tasks. In this work, we use this training strategy to derive learning rules for three substantially different neural architectures. Inference in these models is done by either truncated gradient descent or variational mean-field iterations. In our experiments, we found that the training methods outperform the Contrastive Divergence learning algorithm. Moreover, the training methods can easily handle missing values in the training data itself during learning. We demonstrate the performance of this learning scheme and the three models we introduce on one artificial and two real-world data sets.

Keywords: neural networks, energy-based models, time-series, missing values, optimization

1. Introduction

Many interesting data sets in fields like meteorology, finance, and physics, are high dimensional time-series. High dimensional time-series are also used in applications like speech recognition, motion capture and handwriting recognition. To make optimal use of such data, it is often necessary to impute missing values. Missing values can for example occur due to noise or malfunctioning sensors. Unfortunately, imputing missing values can be a very challenging task when the time series of interest are generated by complex non-linear processes.

Simple techniques like nearest neighbour interpolation treat the data as independent and ignore temporal dependencies. Linear, polynomial and spline-based interpolation techniques tend to fail when variables are missing for extended periods of time. It appears that more complicated models are needed to make good predictions about missing values in high dimensional time-series.

Given a set of observed variables, one can try to define a function that returns a set of predictions for the values that are missing. Models of this type belong to the discriminative family and are for example linear regression, support vector machines and multi-layer perceptrons (Bishop, 2006). Neural networks are interesting candidates for time-series tasks because their connection structure can be designed to capture prior beliefs about the temporal dependencies. Examples of neural networks that are able to deal with temporal sequences are recurrent neural networks and one-

dimensional convolutional neural networks (Waibel et al., 1989). However, most models of the discriminative type assume that the ordering of known and unknown variables is fixed. It is not always clear how to use them if a variable that was known for one data point has to be predicted for another and vice versa.

Nonetheless, there has been some work on training neural networks for missing value recovery in a discriminative way. Nelwamondo et al. (2007) trained autoencoder neural networks to impute missing values in non-temporal data. They used genetic algorithms to insert missing values that maximized the performance of the network. Unfortunately it is not straightforward to apply this method to high dimensional time-series as the required models would be too large for genetic algorithms to remain computationally feasible. Gupta and Lam (1996) trained neural networks for missing value imputation by using some of the input dimensions as input and the remaining ones as output. This requires many neural networks to be trained and limits the available datapoints for each network to those without any missing input dimensions. This method is especially difficult to apply to high dimensional data with many missing dimensions. Convolutional neural networks have been trained to restore images (Jain et al., 2007) in a supervised way but in that work the task was not to impute missing values but to undo the effects of a contaminating process in a way that is more similar to denoising.

At first sight, generative models appear to be a more natural approach to deal with missing values. Probabilistic graphical models (Koller and Friedman, 2009) have often been used to model time-series. Examples of probabilistic graphical models for time-series are Hidden Markov Models (HMM) and linear dynamical systems. For simple tractable models like these, conditional probabilities can be computed analytically. Unfortunately, these simple models have trouble with the long range dependencies and nonlinear structures in many of the more interesting data sets. HMMs have trouble modelling data that is the result of multiple underlying processes because only a single hidden state variable is used. The number of states that is required to model information about the past, grows exponentially as a function of the number of bits to represent. More complicated directed graphical models often suffer from the so-called *explaining away* phenomenon (Pearl, 1988).

Undirected graphical models (also known as Markov Random Fields) have been used as well but tend to be intractable. An example of an intractable model for high dimensional non-linear time series is the Conditional Restricted Boltzmann Machines (CRBMs; Taylor et al. 2007), which was used to reconstruct motion capture data.

There are some less conventional approaches to model nonlinear time series in a generative way as well. A combination of the EM algorithm and the Extended Kalman Smoother can be used to train certain classes of nonlinear dynamical systems (Ghahramani and Roweis, 1999). The difficulty with this approach is that fixed radial basis functions need to be used for approximating the nonlinearities to keep the model tractable. It is not clear how these would scale to higher dimensional state spaces where radial basis functions become exponentially less effective.

Non-parametric models like the Gaussian process latent variable model (Lawrence, 2003) have also been used to develop models for sequential tasks like synthesizing and imputing human motion capture data. A continuation of this work is the Gaussian Process Dynamical Model (Wang et al., 2008). While models of this type tend to display nice generalization properties for small data sets, their application to larger data sets is limited because of the need to invert a great number of kernel matrices that grow cubicly with the number of data points. There has been some work on improving the computational efficiency of these models by introducing sparsity (Lawrence, 2007) but parametric graphical models tend to remain more practical for larger data sets.

We argue that while graphical models are a natural approach to time-series modelling, training them probabilistically is not always the best strategy when the goal is to use them for missing value imputation, especially if the model is intractable. The energy-based framework (LeCun and Huang, 2005; LeCun et al., 2006) permits a discriminative view of graphical models that has a couple of advantages over maximum likelihood learning, especially for a task like missing-value imputation.

- By trying to model the whole joint distribution of a data set, a large part of the flexibility of generative models is used to capture relationships that might not be necessary for the task of interest. A model might put too much effort into assigning low likelihoods to types of data that are very different to the patterns of observed values that will be encountered during the imputation task. An energy model with a deterministic inference method can make predictions that are directly optimized for the task of interest itself.

  Let's for example take an image inpainting task where it is known that the corruption often occurs in the form of large gaps of neighbouring pixels that are missing. In this case, no information from neighbouring values can be used to reconstruct the missing values in the center of a gap. A generative model might have put too much emphasis on the correlations between neighbouring pixels and not be able to efficiently use non-local information. A model that has been trained discriminatively to deal with this type of data corruption would not have this problem. Its parameters have only been optimized to learn the correlations that were relevant for the task without trying to learn the entire distribution over the data.

- Since the normalization constant of many generative models is intractable, inference needs to be done with methods like sampling or variational inference. Deterministic models circumvent this problem.

- Training is intractable as well for many generative graphical models and the design of algorithms that approximate maximum likelihood learning well is still an active area of research (Hyvärinen, 2005; Tieleman, 2008; Tieleman and Hinton, 2009; Desjardins et al., 2010; Brakel et al., 2012). Some of the popular training algorithms like Contrastive Divergence only work well if it is easy to obtain samples from certain groups of the variables in the model.

There are some generative architectures that can handle sequential data with non-linear dependencies. Certain types of Dynamical Factor Graphs (Mirowski and LeCun, 2009) are still tractable when the energy function is designed in such a way that the partition function is remains constant. Another tractable non-linear dynamical system is based on a combination of a recurrent neural network and the neural autoregressive distribution estimator (Larochelle and Murray, 2011; Boulanger-Lewandowski et al., 2012). We will discuss Dynamical Factor Graphs and the generative recurrent neural network model in more detail in Section 7.1 and compare our models with a version of the latter. Overall, however, discriminative energy-based models allow for a broader class of possible models to be applied to missing value imputation while maintaining tractability.

The idea of training Markov Random Fields in a discriminative way by using a simple deterministic inference procedure is not new and has been used in image and natural language processing. In image processing, the inpainting (Bertalmío et al., 2000) or denoising (Barbu, 2009) of pictures are thoroughly studied problems. Barbu (2009) proposed Active Random Fields for denoising images. Active random fields are Fields of Experts (Roth and Black, 2005) that are trained by doing inference with a couple of iterations of gradient descent. The model parameters are optimized to make

the gradient descent inference procedure more efficient. In a more recent paper (Domke, 2012), this approach was extended to more advanced optimization methods like heavy ball (Polyak, 1964) and BFGS. In a similar fashion, gradients have been computed through message passing to train Conditional Random Fields more efficiently (Domke, 2011; Stoyanov et al., 2011). In this paper, we extend their approaches to models for time-series and missing value imputation. To our knowledge, models that were used for image inpainting were either trained in a probabilistic way, or to do denoising. We show that models can be trained for imputation directly and that the approach is not limited to gradient based optimization. It can also be applied to models for which inference is done using the mean-field method. Furthermore, we also show that quite complex models with recurrent dependencies, that would be very difficult to train as probabilistic models, can be learned this way.

The first model we propose is based on a convolution over the data that is coupled to a set of hidden variables. The second model is a recurrent neural network that is coupled to a set of hidden variables as well. In both of these models, inference is done with gradient descent. The third model is a Markov Random Field with distributed hidden state representations for which the inference procedure consists of mean-field iterations.

1.1 Training for Missing Value Imputation

Given a sequence $\mathbf{V}$, represented as a matrix of data vectors $\{\mathbf{v}_1, \cdots, \mathbf{v}_T\}$, let Ω be the set of tuples of indices (i, j) that point to elements of data vectors that have been labelled as missing. For real-valued data, a sound error function to minimize, is the sum of squared errors between the values we predicted $\hat{\mathbf{V}}$ and the actual values $\mathbf{V}$:

$$L = \frac{1}{2} \sum_{(i,j)\in\Omega} (\mathbf{V}_{ij} - \hat{\mathbf{V}}_{ij})^2.$$

Note that this loss function is only defined for a single data sequence. Since Ω will be sampled from some distribution, the actual objective that is minimized during training is the expectation of the sum squared error under a distribution over the missing values as defined for N_{data} sequences by

$$O = \frac{1}{2} \sum_{n=1}^{N_{\text{data}}} \sum_{\Omega} P(\Omega) \sum_{(i,j)\in\Omega} \left(\mathbf{V}(n)_{ij} - \hat{\mathbf{V}}(n)_{ij}\right)^2. \tag{1}$$

All our models will be trained to minimize this objective.

The selection of $P(\Omega)$ during training is task dependent and should reflect prior knowledge about the structure of the missing values. If it is known that missing values occur over multiple adjacent time steps for example, this can be reflected in the choice of $P(\Omega)$. For tasks that contain missing values due to malfunctioning sensors or asynchronous sampling rates, this is pattern likely to be present. We expect that a good choice of $P(\Omega)$ is important but that the objective is robust to $P(\Omega)$ being somewhat imprecise.

1.2 Energy-Based Models and Optimization Based Learning

The models in this paper are inspired by the energy-based framework (LeCun et al., 2006). An Energy-Based Model (EBM) defines a function $E(\cdot)$ that maps a set of observations $\mathbf{V}$ to an energy value. This energy value represents a measure of the 'goodness' of a configuration of the input variables. Most models for regression and classification can be seen as energy models. For a

classifier for example, the variables might be the pixels of an image and a binary class label. A well trained model should assign a lower energy to the image when it is combined with the correct class label than when it is combined with the incorrect one. Inference in energy-based models is done by minimizing the energy function and predictions are defined as

$$\hat{\mathbf{V}}_{(i,j)\in\Omega} \leftarrow \underset{\mathbf{V}_{(i,j)\in\Omega}}{\arg\min} E(\mathbf{V};\theta), \tag{2}$$

where θ is the set of model parameters. The set Ω contains the indices of the variables to perform the minimization over.[1]

In many cases, one wants to solve the optimization problem in Equation 2 with respect to a large number of variables and inference is not as simple any more as in the example with the binary classifier. When the variables of interest are continuous, this optimization problem can be solved with gradient-based optimization methods (or as we will show coordinate descent) but unfortunately it can take many iterations to reach the nearest local minimum of the energy function. In this paper, we use a strategy advocated by Barbu (2009), Domke (2012) and Stoyanov et al. (2011) in which the optimization algorithm has become part of the model. The optimizer is not necessarily run until convergence and a prediction is now given by

$$\hat{\mathbf{V}}_{(i,j)\in\Omega} \leftarrow \underset{\mathbf{V}_{(i,j)\in\Omega}}{\text{opt-alg}}\ E(\mathbf{V};\theta).$$

To train models of this type and find good values for θ, there will be two levels of optimization involved. Firstly, inference is done by minimizing the energy function. Subsequently, gradients with respect to some loss functional are computed through the energy minimization procedure. These gradients are used to optimize the model parameters θ with a method like stochastic gradient descent.

Note that the objective function in Equation 1 is defined in terms of the predicted missing variables. A strict adherence to the energy-based learning framework would require us to use a loss function that is only defined in terms of energy values. Common loss functions for energy-based learning are the log likelihood, the generalized perceptron rule and functions that maximize a margin between the energy values of the desired and undesired configurations of variables (LeCun and Huang, 2005). Most of those loss functions contain a contrastive term that requires one to identify some specific 'most offending' points in the energy landscape that may be difficult to find when the energy landscape is non-convex. The method we chose to use essentially circumvents this problem by transforming the loss into one for which the contrastive term is analytically tractable.

To model complex processes, it is common practice to introduce latent or 'hidden' variables that represent interactions between the observed variables. This leads to an energy function of the form $E(\mathbf{V},\mathbf{H})$, where $\mathbf{H}$ are the hidden variables, which need to be marginalized out to obtain the energy value for $\mathbf{V}$. To discriminate between the value $E(\mathbf{V},\mathbf{H})$ and the sometimes intractable value $E(\mathbf{V})$ we will refer to the former as the energy and the latter as the *free* energy. This summation (or integration) over hidden variables can be greatly simplified by designing models such that the hidden variables are conditionally independent given an observation. A model that satisfies this independence condition is the Restricted Boltzmann Machine (RBM) (Freund and Haussler, 1994; Hinton, 2002), which is often used to build deep belief networks (Hinton et al., 2006).

1. We used Ω again to signify the search space to clarify the role the missing values will play later on.

The first two models we will describe have a tractable free energy function $E(\mathbf{V})$ and we chose to use gradient descent to optimize it. The third model has no tractable free energy $E(\mathbf{V})$ and we chose to use coordinate descent to optimize a variational bound on the free energy instead.

2. The Convolutional Energy-Based Model

We will call the first model the Convolutional Energy-Based Model (CEBM). The CEBM has an energy function that is defined as a one dimensional convolution over a sequence of data combined with a quadratic term and is given by

$$E(\mathbf{V},\mathbf{H}) = \sum_{t=1}^{T} \left(\frac{\|\mathbf{v}_t - \mathbf{b}_v\|^2}{2\sigma^2} - \mathbf{h}_t^\mathsf{T} \mathbf{g}_{\text{conv}}(\mathbf{V},t;\mathbf{W}) - \mathbf{h}_t^\mathsf{T} \mathbf{b}_h \right), \tag{3}$$

where $\mathbf{b}_v$ is a vector with biases for the visible units and $\mathbf{H}$ is a set of binary hidden units. The function $\mathbf{g}_{\text{conv}}(\cdot)$ is defined by

$$\mathbf{g}_{\text{conv}}(\mathbf{V},t;\mathbf{W}) = \mathbf{W}[\mathbf{v}_{t-k} \oplus \cdots \oplus \mathbf{v}_t \oplus \cdots \oplus \mathbf{v}_{t+k}],$$

for $k < t < T-k$ and equal to zero for all other values of t. The matrix $\mathbf{W}$ contains trainable connection weights and the operator $\oplus$ signifies concatenation. The value k determines the number of time frames that each hidden unit is a function of and σ is a parameter for the standard-deviation of the visible variables which will be assumed to be 1 in all experiments. The set of trainable parameters is given by $\theta = \{\mathbf{W}, \mathbf{b}_v, \mathbf{b}_h\}$. This model has the same energy function as the convolutional RBM (Lee et al., 2009) and the main difference is the way training and inference are done. See Fig. 1a for a graphical representation of the convolutional architecture.

The motivation behind this energy function is similar to that of regular RBM models. While correlations between the visible units are not directly parametrized, the hidden variables allow these correlations to be modelled implicitly. When they are not observed, they introduce dependencies among the visible units. The quadratic visual bias term serves a similar role to that of the mean of a Gaussian distribution and prevents the energy function from being unbounded with respect to $\mathbf{V}$.

To compute the free energy of a sequence $\mathbf{V}$, we have to sum over all possible values of $\mathbf{H}$. Fortunately, because the hidden units are binary and independent given the output of the function $\mathbf{g}_{\text{conv}}(\cdot)$, the total free energy can efficiently be calculated analytically and is given by

$$E(\mathbf{V}) = \sum_{t}^{T} \left(\frac{\|\mathbf{v}_t - \mathbf{b}_v\|^2}{2\sigma^2} - \sum_j \log\Big(1 + \exp\big(g_{\text{conv}\,j}(\mathbf{V},t;\mathbf{W}) + \mathbf{b}_h\big)\Big) \right). \tag{4}$$

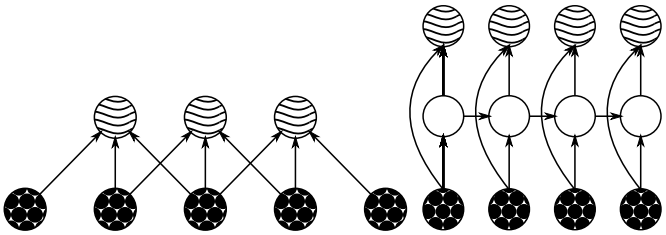
The index j points to the jth hidden unit. See the paper by Freund and Haussler (1994) for a derivation of the analytical sum over the hidden units in Equation 4.

The gradient of the free-energy function with respect to the function value $g_{\text{conv}\,j}(\mathbf{V},t;\mathbf{W})$ is given by the negative sigmoid function:

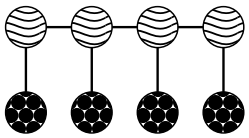
$$\frac{\partial E(\mathbf{V})}{\partial g_{\text{conv}\,j}(\mathbf{V},t;\mathbf{W})} = -\Big(1 + \exp\big(g_{\text{conv}\,j}(\mathbf{V},t;\mathbf{W}) + \mathbf{b}_h\big)\Big)^{-1}.$$

The chain rule can be used to calculate the derivatives with respect to the input of the network. The derivative of the free energy with respect to the input variables is defined as

$$\frac{\partial E(\mathbf{V})}{\partial \mathbf{v}_t} = \frac{\partial E(\mathbf{V})}{\partial \mathbf{g}_{\text{conv}}(\mathbf{V},t;\mathbf{W})} \frac{\partial \mathbf{g}_{\text{conv}}(\mathbf{V},t;\mathbf{W})}{\partial \mathbf{v}_t} + \frac{\mathbf{v}_t - \mathbf{b}_v}{\sigma^2}.$$



(a) Convolutional structure. (b) Recurrent structure.



(c) Undirected structure.

Figure 1: The two model structures that are used in this paper. The wavy circles represent the hidden units, the circles filled with dots the visible units and empty circles represent deterministic functions. The time dimension runs from the left to the right and each circle represents a layer of units.

3. The Recurrent Energy-Based Model

The second model we propose in this paper is the Recurrent Energy-Based Model (REBM). In this model, the energy is based on the dynamics that the data elicit in a non-linear dynamical system. In this case the non-linear dynamical system is instantiated as a Recurrent Neural Network (RNN).

RNNs are interesting models for time-series because they can process sequences of arbitrary length. Unlike Hidden Markov Models, RNNs have hidden states that are high dimensional distributed representations of the previous input patterns.

The energy function of the REBM is again defined by Equation 3 but $\mathbf{g}_{\text{conv}}$ is replaced by

$$\begin{aligned}\mathbf{g}_{\text{rec}}(\mathbf{V},t;\theta_{\text{RNN}}) &= \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{v}_t + \mathbf{b}_r,\\ \mathbf{x}_t &= \tanh(\mathbf{C}\mathbf{x}_{t-1} + \mathbf{D}\mathbf{v}_t + \mathbf{b}_x) \quad 1 < t \leq T,\end{aligned}$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are matrices with network connection parameters and $\mathbf{b}_r$ and $\mathbf{b}_x$ are the bias vectors of, respectively, the output variables and the hidden state variables $\mathbf{X}$. The total set of trainable parameters for this model is given by $\theta = \{\mathbf{b}_h\} \cup \theta_{\text{RNN}}$ with $\theta_{\text{RNN}} = \{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{b}_r,\mathbf{b}_x\}$. See Fig. 1b for a graphical representation of the recurrent architecture.

In most situations, predictions by an RNN only depend on the previous input patterns. For the REBM however, the energy function depends on the full input sequence. This allows predictions to be based on future observations as well.

This model is similar to the Recurrent Temporal Restricted Boltzmann Machine (Sutskever et al., 2008) but in our model the units that define the energy are in a separate layer and the visible variables are not independent given the hidden variables. It is also similar to the Recurrent Neural Network Restricted Boltzmann Machine (Boulanger-Lewandowski et al., 2012) that will be used as a baseline in our experiments.

4. The Discriminative Temporal Boltzmann Machine

The third model is inspired by the work on Deep Boltzmann Machines (Salakhutdinov and Hinton, 2009) and variational inference. In this model, the hidden units are connected over time in a way that is similar to an undirected version of the factorial Hidden Markov Model (Ghahramani and Jordan, 1997). Unlike the factorial Hidden Markov Model however, the hidden variables are not just connected in temporal chains that would be independent given the input (note that this statement is only true for undirected graphs). In this model, every hidden unit at a certain time step t is connected to every hidden unit at time $t+1$ and time $t-1$. This allows the model to use distributed representations that are highly interconnected. The hidden units are again binary and take values from $\{-1,1\}$. The energy of the model is defined by the following equation:

$$E(\mathbf{V},\mathbf{H}) = \sum_{t=1}^{T} \left(\frac{\|\mathbf{v}_t - \mathbf{b}_v\|^2}{2\sigma^2} - \mathbf{h}_{t-1}^\top \mathbf{W}\mathbf{h}_t - \mathbf{h}_t^\top \mathbf{A}\mathbf{v}_t - \mathbf{h}_t^\top \mathbf{b}_h \right),$$

where $\mathbf{h}_0$ is defined to be $\mathbf{0}$. Note that this energy function is very similar to Equation 3, but the convolution has been replaced with a matrix multiplication and there is an additional term that parametrizes correlations between hidden units at adjacent time steps. The structure of the model is shown in Fig. 1c. This model can also be seen as a Deep Boltzmann Machine in which every layer is connected to a different time step of the input sequence. Because this model has the structure of a Boltzmann Machine time-series model and will be trained in a discriminative way, we will refer to the model as the Discriminative Temporal Boltzmann Machine (DTBM). This model is very similar to the one proposed in Williams and Hinton (1991) for discrete data but the way in which we will train it is very different.

4.1 Inference

Since the hidden units of the DTBM are not independent from each other, we are not able to use the free energy formulation from Equation 4. Even when values of the visible units are all known, inference for the hidden units is still intractable. For this reason, we use variational mean-field to minimize the free energy. Compared to other approximate inference algorithms like loopy belief propagation, variational mean-field is relatively computationally efficient as the number of messages to compute is equal to the number of variables rather than their pairwise interactions. In the DTBM the number of pairwise interactions is very large compared to the actual number of variables.

The mean-field approximation can be motivated with ideas from variational inference. In the variational inference framework, inference is cast as an optimization problem (Wainwright and Jordan, 2008). Approximations can now be constructed by relaxing the optimization problem in some way. In the mean-field approach, the optimization is simplified by limiting the set of functions that are used to approximate the function of interest. For probabilistic models, this often means that a simple, tractable, distribution is optimized to be as close as possible to the more complicated, intractable distribution. This is done by optimizing a lower bound on the log likelihood. Optimizing this bound is equivalent to minimization of the Kullback-Leibler divergence between the approximating and target distributions. The simplest way of selecting an approximating distribution, is by dropping all dependencies between the variables. In other words, the approximating distribution takes the form of a product of one-dimensional distributions. This is commonly referred to as *naive mean-field*.

In the original training procedure for the Deep Boltzmann Machine (Salakhutdinov and Hinton, 2009), the free energy is replaced by a variational lower-bound on the log likelihood. Given a set of known variables $\mathbf{x}$ and a set of unknown variables $\mathbf{y}$, this bound can be written as

$$\ln p(\mathbf{x}) \geq \sum_{\mathbf{y}} q(\mathbf{y}|\mathbf{x};\mathbf{u}) \ln p(\mathbf{x},\mathbf{y}) + \mathcal{H}(q),$$

where $q(\cdot)$ is an approximating distribution with parameters $\mathbf{u}$ and $\mathcal{H}(\cdot)$ is the entropy functional.

If the approximating distribution is chosen to be a fully factorized Bernoulli distribution of the form $\prod_{t=1}^{T}\prod_{j=1}^{N_h} q(h_{jt}|u_{jt})$, with $q(h_{jt}=1|u_{jt}) = \frac{1}{2}(u_{jt}+1)$, $q(h_{jt}=-1|u_{jt}) = \frac{1}{2}(1-u_{jt})$ and N_h the number of hidden units, a simple algorithm can be derived that optimizes the set $\mathbf{U} \in (0,1)^{N_h \times T}$ of variational parameters to maximize the bound. While the Deep Boltzmann Machine is originally trained by approximating the free energy component of the gradient with the variational method and the term of the gradient that comes from the partition function with sampling methods, we will only use the variational optimization as our inference algorithm.

Because the distribution will now also be defined over some of the visible units that have been labelled as missing, the variational distribution is augmented with the appropriate number of one-dimensional Gaussian distributions of unit variance of the form $q(v|\hat{u}) = \mathcal{N}(v|\hat{u},1)$. The lower bound now has the following form:

$$\begin{aligned}
\ln p(\mathbf{V}_{\setminus\Omega}) \geq \mathcal{B}(\bar{\mathbf{U}}) = \sum_{t=1}^{T} \Bigg(& -\frac{\|\mathbf{v}_t - \mathbf{b}_v\|^2_{\setminus\Omega}}{2\sigma^2} + \mathbf{u}_{t-1}^{\mathsf{T}}\mathbf{W}\mathbf{u}_t + \mathbf{u}_t^{\mathsf{T}}\mathbf{b}_h \\
& - \frac{\|\hat{\mathbf{u}}_t - \mathbf{b}_v\|^2_{\Omega}}{2\sigma^2} + \sum_{k,i\in\Omega} A_{ki}u_{kt}\hat{u}_{it} + \sum_{k,i\notin\Omega} A_{ki}u_{kt}v_{it} \\
& - \sum_{j=1}^{N_h}\left(\frac{u_{jt}+1}{2}\ln\frac{u_{jt}+1}{2} + \frac{1-u_{jt}}{2}\ln\frac{1-u_{jt}}{2}\right)\Bigg) \\
& + \frac{1}{2}N_{\hat{u}}\ln(2\pi e\sigma^2) - \ln Z(\theta),
\end{aligned}$$

where Ω is the set of indices that point to the variables that have been labelled as missing, $N_{\hat{u}}$ is the number of Gaussian variables to predict (i.e., the number of missing values) and $\bar{\mathbf{U}}$ is the set of all mean field parameters $\mathbf{U} \cup \hat{\mathbf{U}}$. Note that an upper bound on the free energy is now defined as

$$E(\mathbf{V}_{\setminus\Omega}) \leq -\mathcal{B}(\bar{\mathbf{U}}) - \ln Z(\theta).$$

Optimizing this bound will lead to values of the variational parameters that approach a mode of the distribution in a similar way that a minimization of the free energy by means of gradient descent will. Setting the gradient of this bound with respect to the mean-field parameters to zero, leads to the following update equations:

$$\mathbf{u}_t \leftarrow \tanh(\mathbf{W}^{\mathsf{T}}\mathbf{u}_{t+1} + \mathbf{W}\mathbf{u}_{t-1} + \mathbf{b}_h + \mathbf{A}\hat{\mathbf{u}}_t), \tag{5}$$

$$\hat{\mathbf{u}}_t \leftarrow \mathbf{b}_v + \mathbf{A}^{\mathsf{T}}\mathbf{u}_t. \tag{6}$$

Note that only the variables $\hat{\mathbf{u}}$ that correspond to missing values get updated.

Ideally, the variational parameters for the hidden units should be updated in an alternating fashion. The odd units will be mutually independent given the visible variables and the even hidden variables and vice versa. Results about coordinate descent (Bertsekas, 1999) show that the algorithm

is guaranteed to converge to a local minimum of the free energy surface because every individual update achieves the unique minimum for that update and the updates are linearly independent.

The model will be trained to make the variational inference updates more efficient for imputation. The naive mean-field iterations will approach a local mode of the distribution and the values of the parameters $\hat{u}$ can be directly interpreted as predictions. Note that Salakhutdinov and Larochelle (2010) proposed a method that hints in this direction by training a separate model to initialize the mean-field algorithm as well as possible.

5. Computing Loss Gradients

To train the models above, it is necessary to compute the gradient of the loss function with respect to the parameters. The loss gradients of both the models that use gradient descent inference can be computed in a similar way. For these models, only the gradient of the energy is different. Since the DTBM uses a different inference procedure, we describe the computation of its loss gradient in a separate subsection.

5.1 Backpropagation Through Gradient Descent

To train the models that use gradient descent inference (i.e., the convolutional and recurrent models), we backpropagated loss gradients through the gradient descent steps like in Domke (2011). Given an input pattern and a set of indices that point to the missing values, a prediction was first obtained by doing K steps of gradient descent with step size λ on the free energy. Subsequently, the gradient of the mean squared error loss L with respect to the parameters was computed by propagating errors back through the gradient descent steps in holder variables $\bar{\mathbf{V}}$ and $\bar{\theta}$ that in the end contained the gradient of the error with respect to the input variables and the parameters of the models (we use θ as a place holder for both the biases and weights of the models). Note that this procedure is similar to the backpropagation through time procedure for recurrent neural networks. The gradient with respect to the parameters was used to train the models with stochastic gradient descent. Hence, the models were trained to improve the predictions that the optimization procedure came up with directly.

Backpropagation is an application of the chain rule to propagate the error gradient back to the parameters of interest by multiplication of a series of derivatives. A single gradient descent inference step over the input variables is given by

$$\hat{\mathbf{V}}^{k+1} \leftarrow \hat{\mathbf{V}}^k - \lambda \nabla_{\mathbf{V}} E(\hat{\mathbf{V}}^k; \theta). \tag{7}$$

By application of the chain rule, the gradient of the loss with respect to the parameters θ is given by

$$\frac{\partial L}{\partial \theta} = \sum_{k=1}^{K} \frac{\partial L}{\partial \hat{\mathbf{V}}^k} \frac{\partial \hat{\mathbf{V}}^k}{\partial \theta}. \tag{8}$$

Assuming a value of k that is smaller than $K-1$ the gradient of the loss with respect to one of the intermediate states of the variables $\hat{\mathbf{V}}^k$ is given by

$$\frac{\partial L}{\partial \hat{\mathbf{V}}^k} = \frac{\partial L}{\partial \hat{\mathbf{V}}^K} \frac{\partial \hat{\mathbf{V}}^K}{\partial \hat{\mathbf{V}}^{K-1}} \cdots \frac{\partial \hat{\mathbf{V}}^{k+1}}{\partial \hat{\mathbf{V}}^k}.$$

To propagate errors back we need to know $\partial \hat{\mathbf{V}}^{k+1} / \partial \hat{\mathbf{V}}^{k}$. Differentiating Equation 7 with respect to $\mathbf{V}$ gives

$$\frac{\partial \hat{\mathbf{V}}^{k+1}}{\partial \hat{\mathbf{V}}^{k}} = \mathbf{I} - \lambda \frac{\partial^2 E(\hat{\mathbf{V}}^{k}; \theta)}{\partial \mathbf{V} \partial \mathbf{V}^{\mathsf{T}}}.$$

Similarly, to complete the computation in Equation 8, we also need to know $\partial \hat{\mathbf{V}}^{k+1} / \partial \theta$ to propagate the errors back to the model parameters. This partial derivative is given by

$$\frac{\partial \hat{\mathbf{V}}^{k+1}}{\partial \theta} = -\lambda \frac{\partial^2 E(\hat{\mathbf{V}}^{k}; \theta)}{\partial \theta \partial \mathbf{V}^{\mathsf{T}}}.$$

Since we are using gradients of gradients, we need to compute second order derivatives. Both of these second order derivatives are matrices that contain a very large number of values but fortunately there are methods to compute their product with an arbitrary vector efficiently (Domke, 2012; Pearlmutter, 1994). It is never required to explicitly store these values.

One way to compute the product of the second order derivative with a vector is by finite differences using

$$\frac{d\mathbf{f}}{d\mathbf{y}^{\mathsf{T}}}\mathbf{v} \approx \frac{1}{2\varepsilon} \left(\mathbf{f}(\mathbf{y} + \varepsilon \mathbf{v}) - \mathbf{f}(\mathbf{y} - \varepsilon \mathbf{v}) \right).$$

The error of this approximation is $O(\varepsilon^2)$. Another way to compute these values is by means of automated differentiation or the $\mathcal{R}$ operator (Pearlmutter, 1994). In software for automated differentiation like Theano (Bergstra et al., 2010), the required products can be computed relatively efficiently by taking the inner product of the gradient of the energy with respect to the input and the vector to obtain a new cost value. The automated differentiation software can now be used to differentiate this new function again with respect to the input and the parameters.

We found that, even when applied recursively for three steps in single precision, the finite differences approximation was very accurate. For the CEBM, the mean squared difference between the finite differences based loss gradients and the exact gradients was of order 10^{-3}, while the variance of the gradients was of order 10^2. In preliminary experiments, we didn't see any effect of the approximation on the actual performance for this model.

Since the finite difference approximation was generally faster than the exact computation of the second order derivatives, we used it for most of the required quantities. For the REBM we used finite differences with $\varepsilon = 10^{-7}$ in double precision. For the CEBM we found that automatic differentiation for the second order derivative with respect to the parameters was faster so for this quantity we used this method instead of finite differences. The CEBM computations were done on a GPU so we had to use single precision with $\varepsilon = 10^{-4}$.

See Algorithm 1 for more details about the backpropagation through gradient descent procedure. In the algorithm we omitted that all references to the data only concern the missing values to avoid overly complicated notation.

5.2 Backpropagation Through Mean-Field

Computing error gradients for the mean-field based model is possible by backpropagating errors through the mean-field updates. Intuitively, this model can be seen as a bi-directional recurrent neural network with tied weights. The derivatives of the update Equations 5 and 6 are easy to compute using the derivatives of the hyperbolic tangent function and linear operations:

$$\frac{\partial \tanh(\mathbf{W}\mathbf{z})}{\partial \mathbf{z}} = \mathbf{W}^{\mathsf{T}} (1 - \tanh(\mathbf{W}\mathbf{z})^2).$$

Algorithm 1 Compute the error gradient through gradient descent

Initialize $\hat{\mathbf{V}}^0$
for $k = 0$ **to** $K-1$ **do**
  $\hat{\mathbf{V}}^{k+1} \leftarrow \hat{\mathbf{V}}^k - \lambda \nabla_{\hat{\mathbf{V}}} E(\hat{\mathbf{V}}^k; \theta)$
end for
$\bar{\mathbf{V}}^K \leftarrow \nabla L(\hat{\mathbf{V}}^K) = \hat{\mathbf{V}}^K - \mathbf{V}$
$\bar{\theta} \leftarrow \mathbf{0}$
for $k = K-1$ **to** 0 **do**
  $\bar{\theta} \leftarrow \bar{\theta} - \lambda \frac{\partial^2 E(\hat{\mathbf{V}}^k;\theta)}{\partial \theta \partial \mathbf{V}^\top} \bar{\mathbf{V}}^{k+1}$
  $\bar{\mathbf{V}}^k \leftarrow \bar{\mathbf{V}}^{k+1} - \lambda \frac{\partial^2 E(\hat{\mathbf{V}}^k;\theta)}{\partial \mathbf{V} \partial \mathbf{V}^\top} \bar{\mathbf{V}}^{k+1}$
end for
Return $\nabla_\theta L = \bar{\theta}$

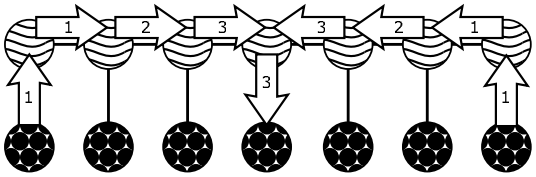


Figure 2: Flow of information due to mean-field updates. The numbers represent the separate iterations at which both the hidden units and the unknown visible units are updated.

However, to make it easy to experiment with the number of mean-field iterations, we used automated differentiation for this.

While the number of gradient descent steps for the convolutional model and the recurrent neural network model can be very low, the number of mean-field iterations has a more profound influence on the behavior of the model. This is because the number of mean-field iterations directly determines the amount of context information that is available to guide the prediction of the missing values. If for example, five iterations of mean-field are used, the activation in the hidden variables at a certain time frame $\mathbf{h}_t$ will be influenced by the five frames of visible variables to the left and right of it and by the known values of $\mathbf{v}_t$; every vector of hidden units now depends on eleven frames of visible units. So a greater number of mean-field iterations increases the range of the dependencies the model can capture. This flow of information is displayed for three iterations in Fig. 2.

Using backpropagation through variational optimization updates is referred to as variational mode learning (Tappen, 2007). In Tappen (2007), Fields of Experts were trained by backpropagating loss function gradients through variational updates that minimized a quadratic upper bound of the loss function. This specific approach would not work for the type of model we defined here.

6. Experiments

We did experiments on three data sets. The first data set consisted of concatenated handwritten digits, the second data set contained marker positions from motion capture recordings and the third data set sensor readings from a mobile robot. The last experiment also investigated the robustness of the models when there were not only missing values in the test set but also in the train data.

To compare our approach with generative methods, we also trained a Convolutional RBM with the same energy function as the Convolutional Energy-Based Model all these data sets. Between these two models, the training method is the only thing that makes them different. Furthermore, we also trained a Recurrent Neural Network Restricted Boltzmann Machine (RNN-RBM; Boulanger-Lewandowski et al. 2012). This model is quite similar to the REBM as it also employs separate sets of deterministic recurrent units and stochastic hidden units. The structure of the RNN-RBM is the same as the architecture in Fig. 1b but the hidden units are connected to the next time step instead of the current one and those connections are undirected. This makes it possible to use Contrastive Divergence learning but also renders the model unable to incorporate future information during inference because the information from the recurrent units is considered to be fixed. Training a recurrent model that incorporates this kind of information with Contrastive Divergence would require something like Hybrid Monte Carlo (Duane et al., 1987). Unfortunately, Hybrid Monte Carlo requires careful tuning of the number and size of leap frog steps and is known to be less efficient than block Gibbs sampling.

For the Energy-Based models we used the same inference procedure as during training. For the Convolutional RBM and RNN-RBM we used mean-field iterations that were run until the MSE changed less than a threshold of 10^{-5}. For the RNN-RBM this was done by initializing the missing values of a single time step with the values of the corresponding dimensions at the previous time-step, running mean-field to update them, passing these new values through the recurrent neural network and repeating this procedure for all remaining time-steps. This procedure proved to be more accurate than Gibbs sampling which was originally used to generate sequences with this model (Boulanger-Lewandowski et al., 2012).

6.1 Concatenated Handwritten Digits

To provide a qualitative assessment of the reconstruction abilities of the models, we used the USPS handwritten digits data set (`http://www.cs.nyu.edu/~roweis/data/usps_all.mat`). While the task we trained the models on is of little practical use, visual inspection of the reconstructions allows for an evaluation of the results that may be more insightful than just the mean squared error with respect to the ground truth.

6.1.1 DATA

The USPS digits data set contains 8-bit grayscale 28×28 pixel images of the numbers '0' through '9'. Each class has 1100 examples. To turn the data into a time-series task, we randomly permuted the order of the digits and concatenated them horizontally. This sequence of 28 dimensional vectors was split into a train set, a validation set and a test set that consisted of respectively 80% and two times 10% of the data.

6.1.2 TRAINING

Good settings of the hyper parameters of the models were found with a random search[2] over 500 points in the parameter space, followed by some manual fine-tuning to find the settings that led to minimal error on the validation set. After this, the models were trained again on both the train

2. The set of hyper parameters included the variances of the Gaussian distributions from which initial weight matrices were sampled, initial learning rates. The numbers of units in each layer were searched over in multiples of 50 up till 300 units.

	Inference iterations	Learning rate	Hidden units
CEBM	3	.005	300
REBM	5	.001	50
DTBM	15	.0005	500
Conv. RBM	N/A	.0005	300
RNN-RBM	N/A	1.8	100

Table 1: Parameter settings for training the models on the USPS data. The learning rates were always divided by the lengths of the sequences during training.

and the validation data with these settings. Since the CEBM, DTBM and Convolutional RBM are very parallel in nature, we used a GPU for their simulations. All models were trained for $100,000$ epochs on randomly selected mini batches of 100 frames. Linearly decaying learning rates were used. Table 1 shows the settings of the hyper parameters. After some preliminary experimentation we found that we got better results for the CEBM by initializing the biases of the hidden units at -2 to promote sparsity. We used a step size of .03 for the gradient descent inference algorithm of the CEBM. The REBM had 200 recurrent units and we used a step size of .2 for the gradient descent inference. The CEBM and the Convolutional RBM had a window size of 7 time frames. The RNN-RBM had 200 recurrent units. Note that the best RNN-RBMs also had more hidden units than the best performing REBMs.

The Convolutional RBM was trained with the Contrastive Divergence algorithm (CD; Hinton 2002). We found that we got better results with this model if we increased the number of CD sampling steps over time. Initially a single CD step was used. After $20,000$ iterations this number was increased to 5, after $50,000$ to 10 and after $75,000$ to 20. We did not find any benefits from stimulating sparsity in this model. For the RNN-RBM we found that a fixed number of 5 CD sampling steps gave the best results.

Missing values were generated as 20 square shaped gaps in every data sequence. The gaps were positioned at uniformly sampled locations. The size of each square was uniformly sampled from the set $\{1,2,\ldots,8\}$. Fig. 3a shows an example of missing values that were generated this way for six sequences. Fig. 3c shows an example of data that has been corrupted by this pattern of missing values.

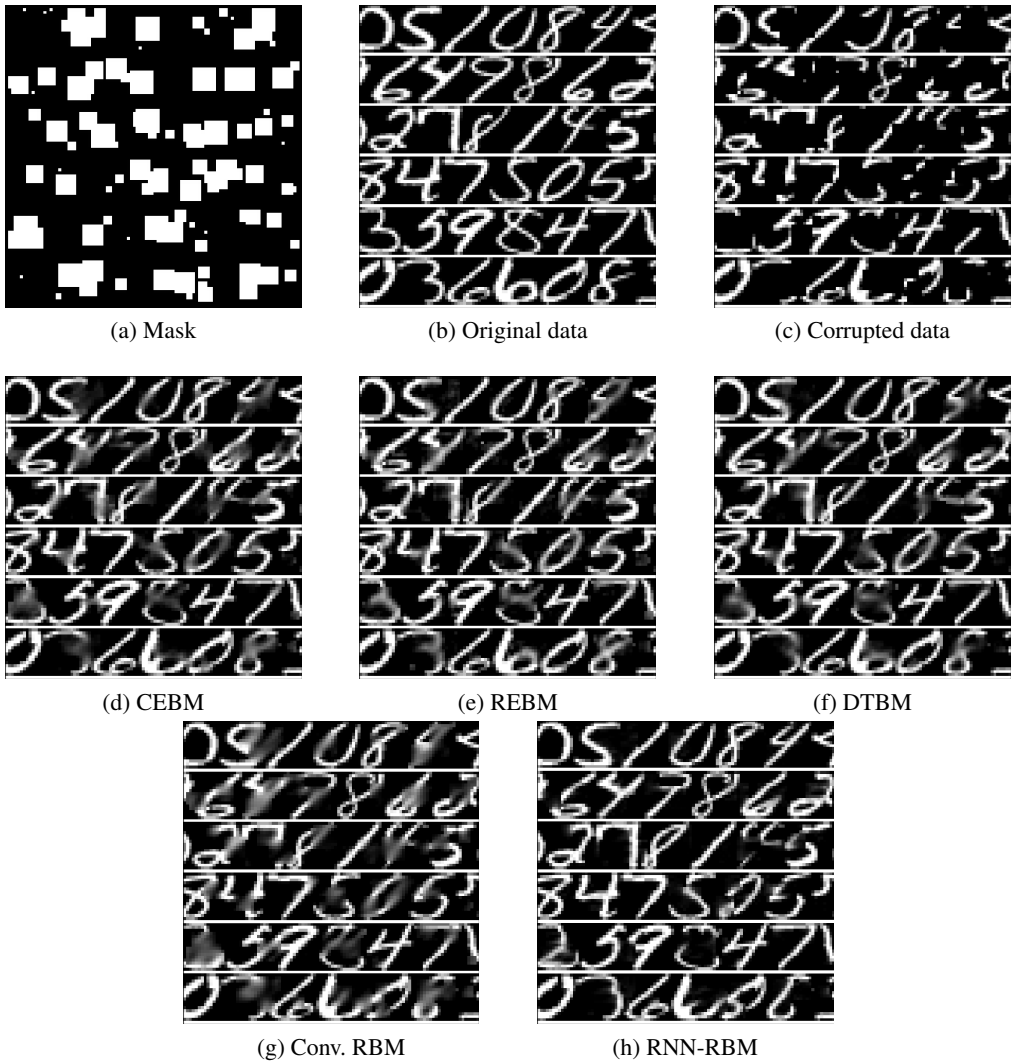
For this experiment, the Energy-Based Models that required missing values during training were provided with missing values from the same distribution that was used to select them for evaluation. To control for the random initialization of the parameters and the randomness induced by stochastic gradient descent, we repeated every experiment 10 times.

6.1.3 RESULTS

Quantitatively, the DTBM achieved the best performance as can be seen in Table 2. The CEBM and REBM performed on a similar level, while the Convolutional RBM and the RNN-RBM performed far worse. Fig. 3 shows how the models reconstructed six damaged sequences from the test data. The reconstructions by the Convolutional RBM in Fig. 3g seem to be of a lower quality than those from the other models and look more blurry. This is consistent with the MSE scores. However, just looking at the MSE scores does not seem to give the full picture as the reconstructions of the CEBM look more smeared out and blurry than those of the REBM even though the MSE scores

	Train	Test
CEBM	.48(.0082)	.49(.0089)
REBM	.47(.01)	.48(.0067)
DTBM	.44(.011)	**.45**(.0088)
Conv. RBM	.66(.011)	.66(.0085)
RNN-RBM	.71(.0056)	.73(.0062)

Table 2: Means and standard deviations of the results in mean squared error on the USPS data.



(a) Mask (b) Original data (c) Corrupted data

(d) CEBM (e) REBM (f) DTBM

(g) Conv. RBM (h) RNN-RBM

Figure 3: Visual representations of the reconstruction of six sequences of handwritten digits. The reconstructions are produced by the CEBM, the REBM, the DTBM, the Convolutional RBM and the RNN-RBM.

of these models are similar. The DTBM provided reconstructions that look similar in quality to those of the REBM. The RNN-RBM has a very bad MSE score while its reconstructions look very sharp when they are correct. Somehow, this model seems to be too sure in cases where it is not able to fill in the data well. This may be due to the fact that the reconstructions are generated on a frame-by frame basis while the deterministic models and the Convolutional RBM can update their predictions from previous iterations. This causes bad predictions to be propagated and possibly amplified. Unfortunately this problem is intrinsic to the choice to keep sampling tractable by treating the recurrent mapping as fixed context information.

6.2 Motion Capture

For the second experiment we applied the models to a motion capture data set. In visual motion capture, a camera records movements of a person wearing a special suit with bright markers attached to it. These markers are later used to link the movements to a skeleton model. Missing values can occur due to lightning effects or occlusion. Motion capture data is high dimensional and generated by a non-linear process. This makes motion capture reconstruction an interesting task for evaluating more complex models for missing value imputation.

In previous work (Taylor et al., 2007), a Conditional Restricted Boltzmann Machine (CRBM) was trained to impute missing values in motion capture as well. For comparison we also train a CRBM but note that a direct comparison of performance is not fair because the CRBM only uses information from a fixed number of previous frames to make predictions.

6.2.1 DATA

The data consisted of three motion capture recordings from 17 marker positions represented as three 49-dimensional sequences of joint angles. The data was down sampled to 30Hz and the sequences consisted of 3128, 438, and 260 frames. The first sequence was used for training, the second for validation and the third for testing. The sequences were derived from a subject who was walking and turning and come from the MIT data set provided by Eugene Hsu.[3] The data was preprocessed by Graham Taylor (Taylor et al., 2007) using parts of Neil Lawrence's Motion Capture Toolbox.

6.2.2 TRAINING

Again, good settings of the hyper parameters of the models were found with a random search on the parameter space, followed by some manual fine-tuning to find the settings that led to minimal error on the validation set. The models were trained on mini batches of 140 frames. Table 3 shows the hyper parameter settings of the models. Note that the best RNN-RBMs had again more hidden units than the best REBMs. Additionally, the inference step sizes of the CEBM and the REBM were both set to .2. The CEBM and the Convolutional RBM had a window size of 15 time frames. All models were trained for $50,000$ iterations. The REBM had 200 recurrent units. To train the Convolutional RBM, single iteration CD training was used during the first $10,000$ epochs. Five iterations of CD were used during the remaining training epochs. The CRBM was trained with single iteration CD. Finally, the RNN-RBM had 200 recurrent units and was trained with 5 CD iterations. All models were trained for a total of $50,000$ epochs.

3. The data set can be found at `http://people.csail.mit.edu/ehsu/work/sig05stf/`.

	Inference iterations	Learning rate	Hidden units
CEBM	3	.5	200
REBM	5	.001	50
DTBM	10	.002	200
CRBM	N/A	.0001	200
Conv. RBM	N/A	.001	200
RNN-RBM	N/A	2.14	100

Table 3: Parameter settings for training the models on the motion capture data. The learning rate values were divided by the lengths of the sequences during training.

The CEBM, REBM and DTBM were again trained by labeling random sets of dimensions as missing. The number of missing dimensions was sampled uniformly. The specific dimensions were then randomly selected without replacement. The duration of the data loss was sampled uniformly between 60 and 125 frames. This adds some bias towards situations in which the same dimensions are missing for a certain duration. This seems to be a sensible assumption in the case of motion capture data and it is an advantage of the training method that it is possible to add this kind of information. To control for the influence of the randomly initialization of the parameters, we repeated every experiment 10 times.

Finally, Nearest neighbour interpolation was done by selecting the frame from the train set with minimal Euclidean distance to the test frame according to the observed dimensions. The distances were computed in the normalized joint angle space.

6.2.3 EVALUATION

To evaluate the models, a set of dimensions was removed from the test data for a duration of 120 frames (4 seconds). This was done for either the markers of the left leg or the markers of the whole upper body (everything above the hip). Because the offset of this gap was chosen randomly and because the CRBM had a stochastic inference procedure, this process was repeated 500 times to obtain average mean squared error values for both the train and the test data. Note that this distribution of missing values was quite different from the one that was used during training.

The CRBM was used in a generative way by conditioning it on the samples it generated at the previous time steps while clamping the observed values and only sampling those that were missing as was done the work by Taylor et al. (2007). Preliminary results showed that this led to similar results to the use of mean-field or minimization of the model's free energy to do inference.

6.2.4 RESULTS

Table 4 shows the mean squared error between the reconstructed dimensions of the data and their actual values. The convolutional and recurrent models clearly outperform the CRBM and nearest neighbour interpolation on the reconstruction of the left leg. The CEBM and the DTBM have the best performance but a comparison of the train and test error scores suggests that the REBM might display better generalization properties. The CRBM seems to suffer most from overfitting. The results of the Convolutional RBM are only slightly worse than the CEBM and better than those of the REBM for the reconstruction of the left leg, but far worse for the upper body where a greater number of variables were missing. The RNN-RBM performed similar to the Convolutional RBM

	Left leg		Upper body	
	Train	Test	Train	Test
CEBM	.18(.0036)	.29(.011)	.47(.023)	.46(.017)
REBM	.22(.0047)	.33(.017)	.47(.014)	.**42**(.014)
DTBM	.17(.0069)	.**28**(.017)	.43(.021)	.45(.011)
CRBM	.25(.0036)	.44(.014)	.51(.023)	.49(.0065)
Conv. RBM	.16(.0038)	.36(.027)	.48(.015)	.68(.035)
RNN-RBM	.18(.0055)	.36(.023)	.45(.017)	.60(.055)
Nearest neighb.	N/A	.45	N/A	.76

Table 4: Means and standard deviations of the results in mean squared error on the motion capture data.

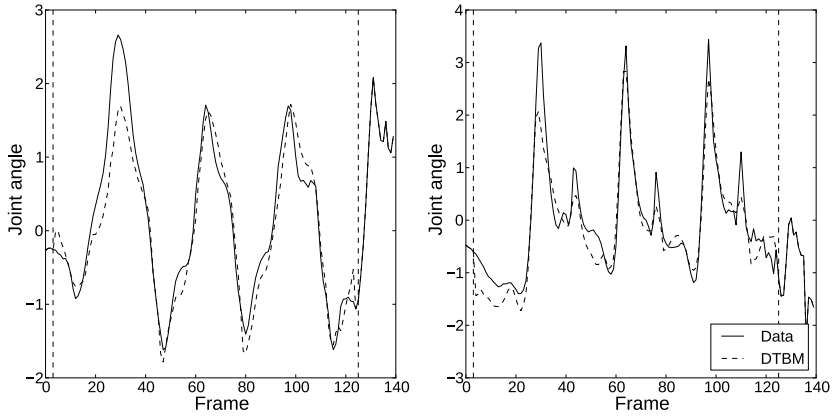


Figure 4: Plots of two dimensions reconstructed by the DTBM next to the actual data. For this sequence the markers of the left leg were missing in the region between the vertical striped lines.

when reconstructing the markers of the missing leg. It performed slightly better at reconstructing the missing upper body than the Convolutional RBM but still a lot worse than the three deterministic models. Fig. 4 shows plots of the predictions made by the DTBM for two of the markers for a sequence from the test data.

6.3 Missing Training Data

So far, all experiments were done by training on data without actual missing values; values were only truly unknown during testing. In practice, a useful model for missing value imputation should also be able to deal with actual missing values in the train set. For generative models, missing values in the train data shouldn't pose a problem because they can be marginalized out. For intractable models like RBMs however, this marginalization can easily become infeasible. For the models we proposed in this paper, missing values in the train data are easily dealt with. During training, the energy is optimized with respect to both the variables for which no ground truth value is available and those that have artificially been labelled as missing. The loss however, is only computed for the

	Inference iterations	Learning rate	Hidden units
CEBM	3	.65	300
REBM	3	.0007	100
DTBM	5	.002	300
Conv. RBM	N/A	.00012	300
RNN-RBM	N/A	.15	50

Table 5: Parameter settings for training the models on the robot data. The learning rates were always divided by the lengths of the sequences during training.

artificially created missing values for which the ground truth is known. A very similar method has been used in earlier work to train neural networks for classification when missing values are present (Bengio and Gingras, 1996).

To see how well our models deal with missing training data, we conducted an additional series of experiments.

6.3.1 DATA

The data we used to investigate the effect of missing training data consists of the measurements of the 24 ultrasound sensors of a SCITOS G5 robot navigating a room (Freire et al., 2009; Frank and Asuncion, 2010). The 5456 sensor readings were sampled at a rate of 9Hz and the robot was following the wall of the room in a clockwise direction, making four trips around the room. We used 80% of the data for training and split the remaining 20% up in a validation set and a test set.

6.3.2 TRAINING

In the first experiment, we trained the models on fully intact training data to get an estimate of the optimal performance the models could achieve on it. In this experiment we also compare the results with those of the Convolutional RBM and the RNN-RBM. In the second experiment, we generated a mask for the whole train set. The mask was divided in regions of 100 frames and at each of these regions a randomly selected set of dimensions was labelled as missing. To train the models, we selected random batches of 100 frames from the train data and selected another set of variables as missing that were not already truly missing in the data. This way, the models never had access to the values that were labelled as missing by the training data mask. To investigate the robustness of the models, we varied the amount of data that was damaged in the train set. In both experiments, the number of dimensions that we pretended to be missing in order to train the models was uniformly sampled from $\{1,\ldots,5\}$. All models were trained for $100,000$ epochs.

The hyper parameter settings were obtained in the same way as in the previous experiments and are displayed in Table 5. Additionally, the Convolutional RBM and CEBM had a window size of 5 and the REBM had 200 recurrent units. The step sizes for the CEBM and REBM were respectively .016 and .7. The CEBM and the Convolutional RBM had a window size of 5 time frames. The Convolutional RBM was trained with the same Contrastive Divergence scheme as in the handwritten digits experiment. The RNN-RBM had 250 recurrent units and was trained with 5 iterations of Contrastive Divergence. Somehow the best performing RNN-RBMs had this time fewer hidden units than the best performing REBMs. We used these settings for all the experiments, regardless of the number of missing training dimensions.

	Train	Test
CEBM	.27	.43
REBM	.41	.39
DTBM	.29	**.34**
Conv. RBM	.53	.47
RNN-RBM	.38	.54

Table 6: Results in mean squared error on the wall robot data without missing dimensions during training.

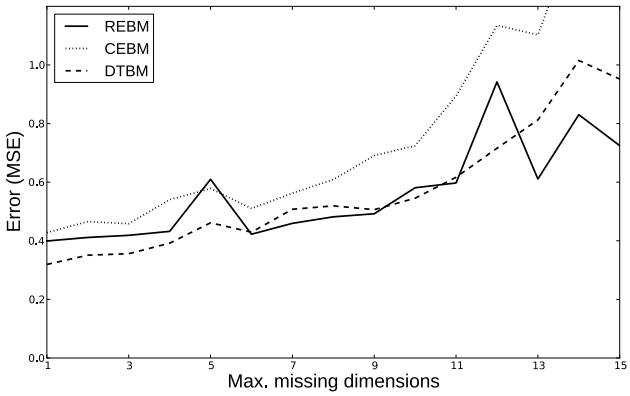


Figure 5: Mean square error as a function of the number of missing dimensions in the train data.

6.3.3 RESULTS

Table 6 shows the results for the experiment without missing train data. The DTBM had the best performance, followed by the REBM, the CEBM, the Convolutional RBM, and the RNN-RBM respectively. Surprisingly, the CEBM has a far better score on the train data than the REBM while its test error is higher. This seems to be a sign of overfitting. The REBM is probably slightly underfitting as its training error was even a little bit higher than its test error. The RNN-RBM obtained a slightly better score on the train data than the REBM but performed a lot worse on the test set.

Fig. 5 shows the error scores for the three energy-based models as a function of the number of missing input dimensions. Obviously, as more dimensions are missing, there is less data available to train on and the error scores of all the models are rising. For the DTBM and REBM, the error seems to increase at a modest pace initially and when just a single value is missing, the performance is similar to the first experiment. The CEBM has more trouble with missing values and is not learning to reconstruct the data well any more after about 10 dimensions are missing. We interpret these results as an upper bound on the error scores one could potentially achieve by doing more extensive hyper parameter tuning for each individual level of data corruption.

7. Discussion

In all the experiments we did, the discriminatively trained models outperformed the baseline methods. The most interesting result is that the CEBM often performed much better than the Convolu-

tional RBM. This indicates that our training strategy is more suited for missing value imputation than the Contrastive Divergence algorithm. The REBM also outperformed the RNN-RBM. While this comparison is less valid because the two models are not entirely the same, it still suggests that our training method also works better for recurrent architectures. The DTBM outperformed all other methods in all of the experiments except for reconstructing the upper body markers of the motion capture data set where the REBM performed better. Given that it is also the easiest model to implement and relatively computationally efficient compared to the other models, this model seems to be the most promising candidate for practical applications.

We didn't show that our method outperforms maximum likelihood learning because Contrastive Divergence is not optimizing the actual likelihood of the model. However, it seems to be the most popular method for training these kind of models when true maximum likelihood learning is intractable. We actually suspect that, compared to other approximate maximum likelihood methods, Contrastive Divergence is still one of the best candidates for missing value imputation because it optimizes the energy landscape locally by pushing up wrong predictions that are near the data itself. When the number of missing values is not too large, inference would start near one of the regions in which the energy landscape has a shape that promotes good predictions. When the number of missing values is too high however, the inference algorithm might start in a region that was not explicitly shaped by the learning algorithm because it was too far away from the data. This might explain the bad performance of the Convolutional RBM when the whole upper body was missing in the motion capture task. The CRBM and (to a lesser extent) the RNN-RBM were more robust in this situation.

7.1 Relation to Other Models

As mentioned before, our training algorithm is based on the work by Domke (2011) with the main difference that we optimize a loss function that only considers a subset of the data variables. Except for the CEBM, our models also have a far more complicated structure and we think that the discriminative training methods are not just more computationally efficient but actually allow us to train models that would otherwise be very difficult to train at all.

Our work is also related to the training of Dynamical Factor Graphs (DFG; Mirowski and LeCun 2009). DFGs are used as generative models, but the way they are trained is more similar to the energy-based learning framework. The types of DFGs that have been studied so far are similar to our REBM model in that they employ both recurrent connections and latent variables. An important difference is that in the REBM the latent variables are not involved in the recurrent part of the model. By constraining the latent states of a DFG to operate under Gaussian noise with fixed covariance, the partition function of the model becomes constant so that a minimization of the energy for a certain state will automatically push up the energy values of all other possible states. The model is not fully probabilistic as it aims to align its maximum a posteriori hidden states with the observed data instead of the sum over all their possible values.

Inference for missing values in DFGs is done with gradient descent to find the minimum energy latent state sequence, ignoring the missing values in the gradient computations. Subsequently, the missing values are predicted as a function of this latent state sequence. There are a couple of important differences with our approach:

- Our models don't aim to provide generative models of the data.

- The CEBM and REBM models use only a limited number of gradient descent updates while for DFGs this minimization is run until convergence.
- In the CEBM and REBM, energy minimization only takes place with respect to the missing values and not with respect to the latent variables which are marginalized out analytically.

It seems that some of the ideas behind DFGs are orthogonal to our approaches and may be combined to design new interesting models.

Recently, a generative model called NADE (Larochelle and Murray, 2011) was proposed in which the mean-field algorithm inspired a model similar to the Restricted Boltzmann Machine (Freund and Haussler, 1994; Hinton, 2002), but with a tractable inference algorithm. In this model, a single iteration of mean-field is used to approximate a set of conditional distributions that are combined into a generative model. Our mean-field based model is substantially different in that it doesn't try to learn a joint distribution over all the variables and that mean-field is also used to estimate the influence of connections between the hidden variables. NADE has been combined with a recurrent neural network to construct a sequential model for note patterns in music (Boulanger-Lewandowski et al., 2012). This model is practically the same as the RNN-RBM we used as a baseline but with the RBM replaced by the NADE model. To our knowledge, NADE has not been applied to continuous data yet, unlike the Recurrent Temporal Restricted Boltzmann Machine (Sutskever et al., 2008), which is more similar to the RNN-RBM.

Adding artificial corruption to the data and training a model to reconstruct it is similar to the way denoising autoencoders are trained (Vincent et al., 2008). The most important difference is that our models only focus on recovery of the missing values given the observed variables and not on reconstructing both.

The way the DTBM model dealt with missing training data in Section 6.3, is very similar to a method for dealing with missing values in neural networks that was presented more than a decade ago (Bengio and Gingras, 1996). In this method, missing values are also filled in by updating them as if they are part of a recurrent neural network. An important difference with our work is that in the work by Bengio and Gingras (1996) the goal was not to predict the missing values themselves, but to perform better on classification tasks when missing values in the inputs are present.

It turns out that the architecture of the DTBM has also been proposed for learning to discriminate between sequences (Williams and Hinton, 1991). In this work, a version of the mean field algorithm that is inspired by simulated annealing is ran until convergence and the training algorithm aims to maximize the lower bound on the likelihood for one class of sequences, while lowering it for the remaining classes. Running mean-field iterations until convergence can make training very inefficient so it might be interesting to see how backpropagation through mean-field performs for tasks of this type.

7.2 Limitations

A downside of our models is that they can take quite long to train. For the CEBM and the DTBM, this problem can be alleviated by using GPU parallelization but unfortunately this is not possible for the REBM. It would probably not have been feasible to train this model with a more generic approach in which the energy optimization would have to be executed until convergence for every training sample during training. However, once training is done, inference for new sequences of data is quite fast. The models took far less time to evaluate than the Convolutional RBM for which mean-field had to be run until convergence to make predictions.

Another downside of our approach is that it introduces new hyper parameters to tune. These are the number of inference iterations and for the CEBM and the REBM also the step size of the gradient descent algorithm. Because of this, we found it easier to find good settings for the DTBM than for the CEBM and REBM. This problem could be solved to some extent by using an optimizer for inference in the CEBM and the REBM that doesn't need a step size parameter. LBFGS has been shown to work quite well for optimization based inference (Domke, 2011) but is a lot more complicated to implement.

The REBM was the most difficult model to train even though it sometimes performed better than the CEBM. It seems to be more sensitive to the initial hyper parameter settings than the other models. As more hidden units were used, these models became more difficult to optimize and this explains why the optimal number of hidden units was generally lower than for the other models. We think this problem occurs because the energy gradient is computed with backpropagation through time over relatively long sequences. The gradients of recurrent neural networks are known to be prone to exponential growth or decay and a single bad gradient can lead to a divergence of the learning algorithm. Larger numbers of hidden units tend to lead to larger values in the gradients, making problematic weight updates more frequent. In preliminary results we found that this problem can be solved to some extent by normalizing the loss gradient before updating the weights. The DTBM might suffer less from this problem because the number of backpropagation steps that are used is smaller.

Since the number of mean-field iterations of the DTBM is directly related to the length of the temporal dependencies it can model, it might not be very suitable for problems in which these dependencies are very long. That being said, this model benefits a lot from parallel computing machinery like GPUs. As the performance of parallel computing hardware increases, the computational time required to model dependencies that are as long as the sequence itself might become more comparable to simulating a regular recurrent neural network. It remains to be seen what kind of effect this has on the quality of the computed gradients.

7.3 Conclusion

We presented a strategy for training models for missing value imputation in high dimensional time-series. The three models we proposed showed promising performance on concatenated digits inpainting, missing marker restoration for motion capture and imputation of values for robot sensors. Our training methods appear to be more suitable for missing value imputation than Contrastive Divergence learning, given similar model architectures. Furthermore, the models could also handle missing values in the training data itself and seem to be relatively robust to these corruptions.

Future work should investigate the performance of different inference methods. Interesting candidates would be variants of loopy belief propagation, expectation propagation and structured mean-field. It would also be interesting to see if models of this type can be designed for other task domains like image inpainting.

Acknowledgments

This article was written under partial support by the FWO Flanders project G.0088.09.

References

A. Barbu. Training an active random field for real-time image denoising. *IEEE Transactions on Image Processing*, 18(11):2451–2462, 2009.

Y. Bengio and F. Gingras. Recurrent neural networks for missing or asynchronous data. *Advances in Neural Information Processing Systems*, pages 395–401, 1996.

J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: A CPU and GPU math compiler in python. In S. van der Walt and J. Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 3–10, 2010.

M. Bertalmío, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *SIGGRAPH*, pages 417–424, 2000.

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

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proceedings of the 29th International Conference on Machine Learning*. Omnipress, 2012.

P. Brakel, S. Dieleman, and B. Schrauwen. Training restricted Boltzmann machines with multi-tempering: Harnessing parallelization. In A. E. P. Villa, W. Duch, P. Érdi, F. Masulli, and G. Palm, editors, *ICANN (2)*, volume 7553 of *Lecture Notes in Computer Science*, pages 92–99. Springer, 2012. ISBN 978-3-642-33265-4.

G. Desjardins, A. C. Courville, Y. Bengio, P. Vincent, and O. Delalleau. Tempered Markov chain Monte Carlo for training of restricted boltzmann machines. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 145–152, 2010.

J. Domke. Parameter learning with truncated message-passing. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 2937–2943, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-1-4577-0394-2. doi: 10.1109/CVPR.2011.5995320.

J. Domke. Generic methods for optimization-based modeling. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 318–326, 2012.

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216–222, 1987.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

A. Freire, G. Barreto, M. Veloso, and A. Varela. Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. In *6th Latin American, Robotics Symposium*, pages 1–6, 2009. doi: 10.1109/LARS.2009.5418323.

Y. Freund and D. Haussler. Unsupervised learning of distributions on binary vectors using two layer networks. Technical report, Santa Cruz, CA, USA, 1994.

Z. Ghahramani and M. I. Jordan. Factorial hidden markov models. *Machine Learning*, 29(2-3): 245–273, 1997.

Z. Ghahramani and S. T. Roweis. Learning nonlinear dynamical systems using an EM algorithm. In *Advances in Neural Information Processing Systems*, pages 599–605. MIT Press, 1999.

A. Gupta and M. S. Lam. Estimating missing values using neural networks. *Journal of the Operational Research Society*, pages 229–238, 1996.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.

V. Jain, J. F. Murray, F. Roth, S. C. Turaga, V. P. Zhigulin, K. L. Briggman, M. Helmstaedter, W. Denk, and H. S. Seung. Supervised learning of image restoration with convolutional networks. In *ICCV*, pages 1–8. IEEE, 2007.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

H. Larochelle and I. Murray. The neural autoregressive distribution estimator. *JMLR: W&CP*, 15: 29–37, 2011.

N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*. MIT Press, 2003. ISBN 0-262-20152-6.

N. D. Lawrence. Learning for larger datasets with the gaussian process latent variable model. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, pages 21–24. Omnipress, 2007.

Y. LeCun and F. Huang. Loss functions for discriminative training of energy-based models. In *Proceedings of the 10-th International Conference on Artificial Intelligence and Statistics*, 2005.

Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. In G. Bakir, T. Hofman, B. Schölkopf, A. Smola, and B. Taskar, editors, *Predicting Structured Data*. MIT Press, 2006.

H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1.

P. Mirowski and Y. LeCun. Dynamic factor graphs for time series modeling. In *Proceedings of the European Conference on Machine Learning*, 2009.

F. V. Nelwamondo, S. Mohamed, and T. Marwala. Missing data: A comparison of neural network and expectation maximisation techniques. *arXiv pre-print*, 2007.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 0-934613-73-7.

B. A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, 6:147–160, 1994.

B. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4:1–17, 1964.

S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *In CVPR*, pages 860–867, 2005.

R. Salakhutdinov and G. E. Hinton. Deep boltzmann machines. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2009.

R. Salakhutdinov and H. Larochelle. Efficient learning of deep boltzmann machines. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2010.

V. Stoyanov, A. Ropson, and J. Eisner. Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, volume 15, pages 725–733, Fort Lauderdale, 2011.

I. Sutskever, G. E. Hinton, and G. W. Taylor. The recurrent temporal restricted boltzmann machine. In *Advances in Neural Information Processing Systems*, volume 21, Cambridge, MA, 2008. MIT Press.

M. F. Tappen. Utilizing variational optimization to learn Markov random fields. In *CVPR*. IEEE Computer Society, 2007.

G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. In *Advances in Neural Information Processing Systems*, volume 19, Cambridge, MA, 2007. MIT Press.

T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the International Conference on Machine Learning*, 2008.

T. Tieleman and G. E. Hinton. Using fast weights to improve persistent contrastive divergence. In *Proceedings of the 26th International Conference on Machine learning*, pages 1033–1040. ACM New York, NY, USA, 2009.

P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1096–1103, New York, NY, USA, 2008. ACM.

A. Waibel, T. Hanazawa, G. E. Hinton, K. Shikano, and K. J. Lang. Phoneme recognition using time-delay neural networks. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37 (3):328–339, 1989.

M. J. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc., Hanover, MA, USA, 2008. ISBN 1601981848, 9781601981844.

J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):283–298, 2008.

C. K. Williams and G. E. Hinton. Mean field networks that learn to discriminate temporally distorted strings. In *Connectionist Models: Proceedings of the 1990 Summer School*, pages 18–22. San Mateo, CA: Morgan Kaufmann, 1991.

Journal of Machine Learning Research 14 (2013) 2799-2835 Submitted 12/12; Revised 6/13; Published 9/13

Belief Propagation for Continuous State Spaces: Stochastic Message-Passing with Quantitative Guarantees

Nima Noorshams NNOORSHAMS@CAL.BERKELEY.EDU
Martin J. Wainwright WAINWRIG@EECS.BERKELEY.EDU
Department of Electrical Engineering & Computer Sciences
University of California Berkeley
Berkeley, CA 94720-1770, USA

Editor: Manfred Opper

Abstract

The sum-product or belief propagation (BP) algorithm is a widely used message-passing technique for computing approximate marginals in graphical models. We introduce a new technique, called stochastic orthogonal series message-passing (SOSMP), for computing the BP fixed point in models with continuous random variables. It is based on a deterministic approximation of the messages via orthogonal series basis expansion, and a stochastic estimation of the basis coefficients via Monte Carlo techniques and damped updates. We prove that the SOSMP iterates converge to a δ-neighborhood of the unique BP fixed point for any tree-structured graph, and for any graphs with cycles in which the BP updates satisfy a contractivity condition. In addition, we demonstrate how to choose the number of basis coefficients as a function of the desired approximation accuracy δ and smoothness of the compatibility functions. We illustrate our theory with both simulated examples and in application to optical flow estimation.

Keywords: graphical models, sum-product for continuous state spaces, low-complexity belief propagation, stochastic approximation, Monte Carlo methods, orthogonal basis expansion

1. Introduction

Graphical models provide a parsimonious yet flexible framework for describing probabilistic dependencies among large numbers of random variables. They have proven useful in a variety of application domains, including computational biology, computer vision and image processing, data compression, and natural language processing, among others. In all of these applications, a central computational challenge is the *marginalization problem*, by which we mean the problem of computing marginal distributions over some subset of the variables. Naively approached, such marginalization problems become intractable for all but toy problems, since they entail performing summation or integration over high-dimensional spaces. The sum-product algorithm, also known as belief propagation (BP), is a form of dynamic programming that can be used to compute exact marginals much more efficiently for graphical models without cycles, known as trees. It is an iterative algorithm in which nodes in the graph perform a local summation/integration operation, and then relay results to their neighbors in the form of messages. Although it is guaranteed to be exact on trees, it is also commonly applied to graphs with cycles, in which context it is often known as loopy BP. For more details on graphical models and BP, we refer the readers to the papers by Kschischang

et al. (2001), Wainwright and Jordan (2008), Aji and McEliece (2000), Loeliger (2004) and Yedidia et al. (2005).

In many applications of graphical models, we encounter random variables that take on continuous values (as opposed to discrete). For instance, in computer vision, the problem of optical flow calculation is most readily formulated in terms of estimating a vector field in $\mathbb{R}^2$. Other applications involving continuous random variables include tracking problems in sensor networks, vehicle localization, image geotagging, and protein folding in computational biology. With certain exceptions (such as multivariate Gaussian problems), the marginalization problem is very challenging for continuous random variables: in particular, the messages correspond to functions, so that they are expensive to compute and transmit, in which case BP may be limited to small-scale problems. Motivated by this challenge, researchers have proposed different techniques to reduce complexity of BP in different applications (Arulampalam et al., 2002; Sudderth et al., 2010; Isard, 2003; Doucet et al., 2001; Ihler and McAllester, 2009; Coughlan and Shen, 2007; Isard et al., 2009; Song et al., 2011; Noorshams and Wainwright, 2012, 2013). For instance, various types of quantization schemes (Coughlan and Shen, 2007; Isard et al., 2009) have been used to reduce the effective state space and consequently the complexity. In another line of work, researchers have proposed stochastic methods inspired by particle filtering (Arulampalam et al., 2002; Sudderth et al., 2010; Isard, 2003; Doucet et al., 2001; Ihler and McAllester, 2009). These techniques are typically based on approximating the messages as weighted particles (Doucet et al., 2001; Ihler and McAllester, 2009), or mixture of Gaussians (Sudderth et al., 2010; Isard, 2003). Other researchers (Song et al., 2011) have proposed the use of kernel methods to simultaneously estimate parameters and compute approximate marginals in a simultaneous manner.

In this paper, we present a low-complexity (efficient) alternative to belief propagation with continuous variables. Our method, which we refer to as stochastic orthogonal series message-passing (SOSMP), is applicable to general pairwise Markov random fields, and is equipped with various theoretical guarantees. As suggested by its name, the algorithm is based on combining two ingredients: orthogonal series approximation of the messages, and the use of stochastic updates for efficiency. In this way, the SOSMP updates lead to a randomized algorithm with substantial reductions in communication and computational complexity. Our main contributions are to analyze the convergence properties of the SOSMP algorithm, and to provide rigorous bounds on the overall error as a function of the associated computational complexity. In particular, for tree-structured graphs, we establish almost sure convergence, and provide an explicit inverse polynomial convergence rate (Theorem 2). For loopy graphical models on which the usual BP updates are contractive, we also establish similar convergence rates (Theorem 3). Our general theory provides quantitative upper bounds on the number of iterations required to compute a δ-accurate approximation to the BP message fixed point, as we illustrate in the case of kernel-based potential functions (Theorem 4).

The reminder of the paper is organized as follows. We begin in Section 2, with the necessary background on the graphical models as well as the BP algorithm. Section 3 is devoted to a precise description of the SOSMP algorithm. In Section 4, we state our main theoretical results and develop some of their corollaries. In order to demonstrate the algorithm's effectiveness and confirm theoretical predictions, we provide some experimental results, on both synthetic and real data, in Section 5. In Section 6, we provide the proofs of our main results, with some of the technical aspects deferred to the appendices.

2. Background

We begin by providing some background on graphical models and the BP algorithm.

2.1 Undirected Graphical Models

Consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consisting of a collection of nodes $\mathcal{V} = \{1, 2, \ldots, n\}$, along with a collection of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. An edge is an undirected pair (u, v), and self-edges are forbidden (meaning that $(u, u) \notin \mathcal{E}$ for all $u \in \mathcal{V}$). For each $u \in \mathcal{V}$, let X_u be a random variable taking values in a space $\mathcal{X}_u$. An undirected graphical model, also known as a Markov random field, defines a family of joint probability distributions over the random vector $X := \{X_u,\ u \in \mathcal{V}\}$, in which each distribution must factorize in terms of local potential functions associated with the cliques of the graph. In this paper, we focus on the case of pairwise Markov random fields, in which case the factorization is specified in terms of functions associated with the nodes and edges of the graph.

More precisely, we consider probability densities p that are absolutely continuous with respect to a given measure μ, typically the Lebesgue measure for the continuous random variables considered here. We say that *p respects the graph structure* if it can be factorized in the form

$$p(x_1, x_2, \ldots, x_n) \propto \prod_{u \in \mathcal{V}} \psi_u(x_u) \prod_{(u,v) \in \mathcal{E}} \psi_{uv}(x_u, x_v). \tag{1}$$

Here $\psi_u : \mathcal{X}_u \to (0, \infty)$ is the node potential function, whereas $\psi_{uv} : \mathcal{X}_u \times \mathcal{X}_v \to (0, \infty)$ denotes the edge potential function. A factorization of this form (1) is also known as *pairwise Markov random field*; see Figure 1 for a few examples that are widely used in practice.

In many applications, a central computational challenge is the computation of the marginal distribution

$$p(x_u) \ := \underbrace{\int_{\mathcal{X}} \cdots \int_{\mathcal{X}}}_{(n-1) \text{ times}} p(x_1, x_2, \ldots, x_n) \prod_{v \in \mathcal{V} \setminus \{u\}} \mu(dx_v) \tag{2}$$

at each node $u \in \mathcal{V}$. Naively approached, this problem suffers from the curse of dimensionality, since it requires computing a multi-dimensional integral over an $(n-1)$-dimensional space. For Markov random fields defined on trees (graphs without cycles), part of this exponential explosion can be circumvented by the use of the BP or sum-product algorithm, to which we turn in the following section.

Before proceeding, let us make a few comments about the relevance of the marginals in applied problems. In a typical application, one also makes independent noisy observations y_u of each hidden random variable X_u. By Bayes' rule, the posterior distribution of X given the observations $y = (y_1, \ldots, y_n)$ then takes the form

$$p_{X|Y}(x_1, \ldots, x_n \mid y_1, \ldots, y_n) \propto \prod_{u \in \mathcal{V}} \widetilde{\psi}_u(x_u; y_u) \prod_{(u,v) \in \mathcal{E}} \psi_{uv}(x_u, x_v), \tag{3}$$

where we have introduced the convenient shorthand for the modified node-wise potential functions $\widetilde{\psi}_u(x_u; y_u) := p(y_u \mid x_u)\, \psi_u(x_u)$. Since the observation vector y is fixed and known, any computational problem for the posterior distribution (3) can be reduced to an equivalent problem for a pairwise Markov random field of the form (1), using the given definition of the modified potential

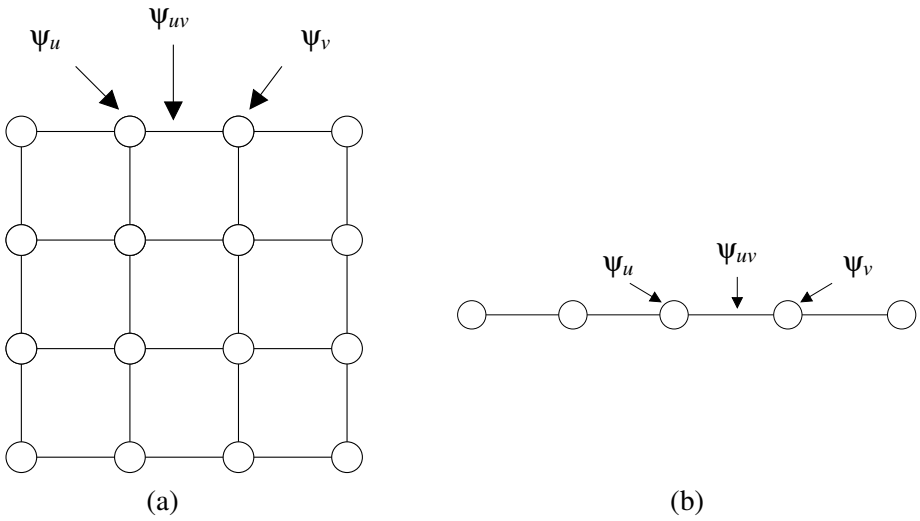


Figure 1: Examples of pairwise Markov random fields. (a) Two-dimensional grid. (b) Markov chain model. Potential functions ψ_u and ψ_v are associated with nodes u and v respectively, whereas potential function ψ_{uv} is associated with edge (u,v).

functions. In addition, although our theory allows for distinct state spaces $\mathcal{X}_u$ at each node $u \in \mathcal{V}$, throughout the remainder of the paper, we suppress this possible dependence so as to simplify exposition.

2.2 Belief Propagation

The BP algorithm, also known as the sum-product algorithm, is an iterative method based on message-passing updates for computing either exact or approximate marginal distributions. For trees (graphs without cycles), it is guaranteed to converge after a finite number of iterations and yields the exact marginal distributions, whereas for graphs with cycles, it yields only approximations to the marginal distributions. Nonetheless, this "loopy" form of BP is widely used in practice. Here we provide a very brief treatment sufficient for setting up the main results and analysis of this paper, referring the reader to various standard sources (Kschischang et al., 2001; Wainwright and Jordan, 2008) for further background.

In order to define the message-passing updates, we require some further notation. For each node $v \in \mathcal{V}$, let $\mathcal{N}(v) := \{u \in \mathcal{V} \mid (u,v) \in \mathcal{E}\}$ be its set of neighbors, and we use $\vec{\mathcal{E}}(v) := \{(v \to u) \mid u \in \mathcal{N}(v)\}$ to denote the set of all directed edges emanating from v. We use $\vec{\mathcal{E}} := \cup_{v \in \mathcal{V}} \vec{\mathcal{E}}(v)$ to denote the set of all directed edges in the graph. Let $\mathcal{M}$ denote the set of all probability densities (with respect to the base measure μ) defined on the space $\mathcal{X}$—that is

$$\mathcal{M} \;=\; \Big\{ m : \mathcal{X} \to [0,\infty) \,\Big|\, \int_{\mathcal{X}} m(x)\mu(dx) = 1 \Big\}.$$

The messages passed by the BP algorithm are density functions, taking values in the space $\mathcal{M}$. More precisely, we assign one message $m_{v \to u} \in \mathcal{M}$ to every directed edge $(v \to u) \in \vec{\mathcal{E}}$, and we denote the

collection of all messages by $m = \{m_{v\to u},\ (v \to u) \in \vec{\mathcal{E}}\}$. Note that the full collection of messages m takes values in the product space $\mathcal{M}^{|\vec{\mathcal{E}}|}$.

At an abstract level, the BP algorithm generates a sequence of message densities $\{m^t\}$ in the space $\mathcal{M}^{|\vec{\mathcal{E}}|}$, where $t = 0, 1, 2\ldots$ is the iteration number. The update of message m^t to message m^{t+1} can be written in the form $m^{t+1} = \mathcal{F}(m^t)$, where $\mathcal{F} : \mathcal{M}^{|\vec{\mathcal{E}}|} \to \mathcal{M}^{|\vec{\mathcal{E}}|}$ is a non-linear operator. This global operator is defined by the local update operators[1] $\mathcal{F}_{v\to u} : \mathcal{M}^{|\vec{\mathcal{E}}|} \to \mathcal{M}$, one for each directed edge of the graph, such that $m^{t+1}_{v\to u} = \mathcal{F}_{v\to u}(m^t)$.

More precisely, in terms of these local updates, the BP algorithm operates as follows. At each iteration $t = 0, 1, \ldots$, each node $v \in \mathcal{V}$ performs the following steps:

- for each one of its neighbors $u \in \mathcal{N}(v)$, it computes $m^{t+1}_{v\to u} = \mathcal{F}_{v\to u}(m^t)$.
- it transmits message $m^{t+1}_{v\to u}$ to neighbor $u \in \mathcal{N}(v)$.

In more detail, the message update takes the form

$$\underbrace{[\mathcal{F}_{v\to u}(m^t)](\cdot)}_{m^{t+1}_{v\to u}(\cdot)} := \kappa \int_{\mathcal{X}} \Big\{ \psi_{uv}(\cdot, x_v)\, \psi_v(x_v) \prod_{w\in\mathcal{N}(v)\setminus\{u\}} m^t_{w\to v}(x_v) \Big\} \mu(dx_v), \tag{4}$$

where κ is a normalization constant chosen to enforce the normalization condition

$$\int_{\mathcal{X}} m^{t+1}_{v\to u}(x_u)\, \mu(dx_u) \;=\; 1.$$

By concatenating the local updates (4), we obtain a global update operator $\mathcal{F} : \mathcal{M}^{|\vec{\mathcal{E}}|} \to \mathcal{M}^{|\vec{\mathcal{E}}|}$, as previously discussed. The goal of belief propagation message-passing is to obtain a *fixed point*, meaning an element $m^* \in \mathcal{M}^{|\vec{\mathcal{E}}|}$ such that $\mathcal{F}(m^*) = m^*$. Under mild conditions, it can be shown that there always exists at least one fixed point, and for any tree-structured graph, the fixed point is unique.

Given a fixed point m^*, each node $u \in \mathcal{V}$ computes its marginal approximation $\tau^*_u \in \mathcal{M}$ by combining the local potential function ψ_u with a product of all incoming messages as

$$\tau^*_u(x_u) \;\propto\; \psi_u(x_u) \prod_{v\in\mathcal{N}(u)} m^*_{v\to u}(x_u). \tag{5}$$

Figure 2 provides a graphical representation of the flow of the information in these local updates. For tree-structured (cycle-free) graphs, it is known that BP updates (4) converge to the unique fixed point in a finite number of iterations (Wainwright and Jordan, 2008). Moreover, the quantity $\tau^*_u(x_u)$ is equal to the single-node marginal, as previously defined (2). For general graphs, uniqueness of the fixed point is no longer guaranteed (Wainwright and Jordan, 2008); however, the same message-passing updates can be applied, and are known to be extremely effective for computing approximate marginals in numerous applications.

1. It is worth mentioning, and important for the computational efficiency of BP, that $m_{v\to u}$ is only a function of the messages $m_{w\to v}$ for $w \in \mathcal{N}(v)\setminus\{u\}$. Therefore, we have $\mathcal{F}_{v\to u} : \mathcal{M}^{d_v-1} \to \mathcal{M}$, where d_v is the degree of the node v. However, we suppress this local dependence so as to reduce notational clutter.

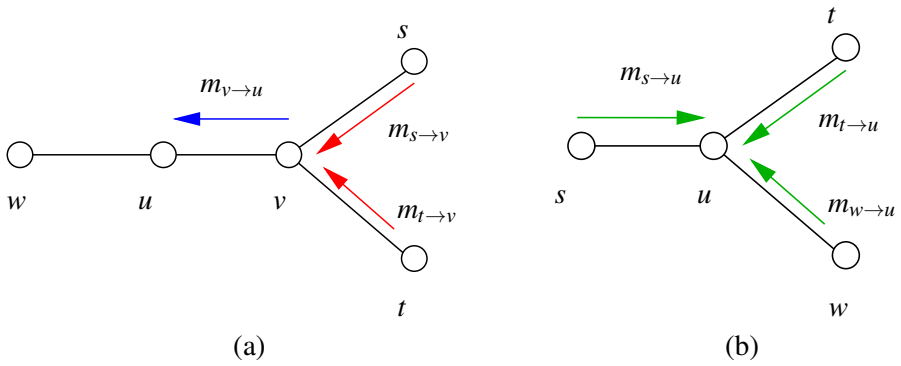


Figure 2: Graphical representation of message-passing algorithms. (a) Node v transmits the message $m_{v\to u} = \mathcal{F}_{v\to u}(m)$, derived from Equation (4), to its neighbor u. (b) Upon receiving all the messages, node u updates its marginal estimate according to (5).

Although the BP algorithm is considerably more efficient than the brute force approach to marginalization, the message update Equation (4) still involves computing an integral and transmitting a real-valued function (message). With certain exceptions (such as multivariate Gaussians), these continuous-valued messages do not have finite representations, so that this approach is computationally very expensive. Although integrals can be computed by numerical methods, the BP algorithm requires performing many such integrals *at each iteration*, which becomes very expensive in practice.

3. Description of the Algorithm

We now turn to the description of the SOSMP algorithm. Before doing so, we begin with some background on the main underlying ingredients: orthogonal series expansion, and stochastic message updates.

3.1 Orthogonal Series Expansion

As described in the previous section, for continuous random variables, each message is a density function in the space $\mathcal{M} \subset L^2(\mathcal{X};\mu)$. We measure distances in this space using the usual L^2 norm $\|f-g\|_2^2 := \int_{\mathcal{X}}(f(x)-g(x))^2\mu(dx)$. A standard way in which to approximate functions is via orthogonal series expansion. In particular, let $\{\phi_j\}_{j=1}^{\infty}$ be an orthonormal basis of $L^2(\mathcal{X};\mu)$, meaning a collection of functions such that

$$\underbrace{\int_{\mathcal{X}} \phi_i(x)\phi_j(x)\mu(dx)}_{:=\langle\phi_i,\phi_j\rangle_{L^2}} = \begin{cases} 1 & \text{when } i=j \\ 0 & \text{otherwise.} \end{cases}$$

Any function $f \in \mathcal{M} \subset L^2(\mathcal{X};\mu)$ then has an expansion of the form $f = \sum_{j=1}^{\infty} a_j\phi_j$, where $a_j = \langle f, \phi_j\rangle_{L^2}$ are the basis expansion coefficients.

Of course, maintaining the infinite sequence of basis coefficients $\{a_j\}_{j=1}^{\infty}$ is also computationally intractable, so that any practical algorithm will maintain only a finite number r of basis coefficients. For a given r, we let $\widehat{f}_r \propto \left[\sum_{j=1}^{r} a_j \phi_j\right]_+$ be the approximation based on the first r coefficients. (Applying the operator $[t]_+ = \max\{0,t\}$ amounts to projecting $\sum_{j=1}^{r} a_j\phi_j$ onto the space of non-negative functions, and we also normalize to ensure that it is a density function.) In using only r coefficients, we incur the *approximation error*

$$\|\widehat{f}_r - f\|_2^2 \stackrel{(i)}{\leq} \|\sum_{j=1}^{r} a_j\phi_j - f\|_2^2 \stackrel{(ii)}{=} \sum_{j=r+1}^{\infty} a_j^2, \tag{6}$$

where inequality (i) uses non-expansivity of the projection, and step (ii) follows from Parseval's theorem. Consequently, the approximation error will depend both on

- how many coefficients r that we retain, and
- the decay rate of the expansion coefficients $\{a_j\}_{j=1}^{\infty}$.

For future reference, it is worth noting that the local message update (4) is defined in terms of an integral operator of the form

$$f(\cdot) \;\mapsto\; \int_{\mathcal{X}} \psi_{uv}(\cdot, y)\, f(y)\, \mu(dy). \tag{7}$$

Consequently, whenever the edge potential function ψ_{uv} has desirable properties—such as differentiability and/or higher order smoothness—then the messages also inherit these properties. With an appropriate choice of the basis $\{\phi_j\}_{j=1}^{\infty}$, such properties translate into decay conditions on the basis coefficients $\{a_j\}_{j=1}^{\infty}$. For instance, for α-times differentiable functions expanded into the Fourier basis, the Riemann-Lebesgue lemma guarantees that the coefficients a_j decay faster than $(1/j)^{2\alpha}$. We develop these ideas at greater length in the sequel.

3.2 Stochastic Message Updates

In order to reduce the approximation error (6), the number of coefficients r needs to be increased (as a function of the ultimate desired error δ). Since increases in r lead to increases in computational complexity, we need to develop effective reduced-complexity methods. In this section, we describe (at a high-level) how this can be done via a stochastic version of the BP message-passing updates.

We begin by observing that message update (4), following the appropriate normalization, can be cast as an expectation operation. This equivalence is essential, because it allows us to obtain unbiased approximations of the message update using stochastic techniques. In particular, for every directed edge $(v \to u)$ let us define

$$\Gamma_{uv}(\cdot, y) \;:=\; \frac{\psi_{uv}(\cdot, y)}{\int_{\mathcal{X}} \psi_{uv}(x,y)\, \mu(dx)}, \quad \text{and} \quad \beta_{v\to u}(y) \;:=\; \psi_v(y) \int_{\mathcal{X}} \psi_{uv}(x,y)\, \mu(dx), \tag{8}$$

the normalized compatibility function and the marginal potential weight respectively. By construction, for each y, we have $\int_{\mathcal{X}} \Gamma_{uv}(x,y)\mu(dx) = 1$.

Lemma 1 *Given an input collection of messages m, let Y be a random variable with density proportional to*

$$[p_{v\to u}(m)](y) \propto \beta_{v\to u}(y) \prod_{w\in\mathcal{N}(v)\setminus\{u\}} m_{w\to v}(y). \tag{9}$$

Then the message update Equation (4) *can be written as*

$$[\mathcal{F}_{v\to u}(m)](\cdot) = \mathbb{E}_Y\big[\Gamma_{uv}(\cdot, Y)\big].$$

Proof Let us introduce the convenient shorthand $M(y) = \prod_{w\in\mathcal{N}(v)\setminus\{u\}} m_{w\to v}(y)$. By definition (4) of the message update, we have

$$[\mathcal{F}_{v\to u}(m)](\cdot) = \frac{\int_{\mathcal{X}} \big(\psi_{uv}(\cdot, y)\,\psi_v(y)\,M(y)\mu(dy)}{\int_{\mathcal{X}}\int_{\mathcal{X}} \big(\psi_{uv}(x,y)\,\psi_v(y)M(y)\big)\,\mu(dy)\,\mu(dx)}.$$

Since the integrand is positive, by Fubini's theorem (Durrett, 1995), we can exchange the order of integrals in the denominator. Doing so and simplifying the expression yields

$$[\mathcal{F}_{v\to u}(m)](\cdot) = \int_{\mathcal{X}} \underbrace{\frac{\psi_{uv}(\cdot, y)}{\int_{\mathcal{X}}\psi_{uv}(x,y)\mu(dx)}}_{\Gamma_{uv}(\cdot,y)} \underbrace{\frac{\beta_{v\to u}(y)M(y)}{\int_{\mathcal{X}}\beta_{v\to u}(z)M(z)\mu(dz)}}_{[p_{v\to u}(m)](y)}\,\mu(dy),$$

which establishes the claim. ■

Based on Lemma 1, we can obtain a stochastic approximation to the message update by drawing k i.i.d. samples Y_i from the density (9), and then computing $\sum_{i=1}^k \Gamma_{uv}(\cdot, Y_i)\,/\,k$. Given the non-negativity and chosen normalization of Γ_{uv}, note that this estimate belongs to $\mathcal{M}$ by construction. Moreover, it is an unbiased estimate of the correctly updated message, which plays an important role in our analysis.

3.3 Precise Description of the Algorithm

The SOSMP algorithm involves a combination of the orthogonal series expansion techniques and stochastic methods previously described. Any particular version of the algorithm is specified by the choice of basis $\{\phi_j\}_{j=1}^{\infty}$ and two positive integers: the number of coefficients r that are maintained, and the number of samples k used in the stochastic update. Prior to running the algorithm, for each directed edge $(v\to u)$, we pre-compute the inner products

$$\gamma_{v\to u;j}(y) := \underbrace{\int_{\mathcal{X}} \Gamma_{uv}(x,y)\phi_j(x)\,\mu(dx)}_{\langle\Gamma_{uv}(\cdot,y),\phi_j(\cdot)\rangle_{L^2}}, \qquad \text{for } j = 1,\ldots,r. \tag{10}$$

When ψ_{uv} is a symmetric and positive semidefinite kernel function, these inner products have an explicit and simple representation in terms of its Mercer eigendecomposition (see Section 4.3). In the general setting, these r inner products can be computed via standard numerical integration techniques. Note that this is a fixed (one-time) cost prior to running the algorithm.

SOSMP algorithm:

1. At time $t = 0$, initialize the message coefficients

$$a^0_{v\to u;j} = 1/r \quad \text{for all } j = 1,\ldots,r, \text{ and } (v \to u) \in \vec{\mathcal{E}}.$$

2. For iterations $t = 0, 1, 2, \ldots$, and for each directed edge $(v \to u)$

   (a) Form the projected message approximation $\widehat{m}^t_{w\to v}(\cdot) = \left[\sum_{j=1}^r a^t_{w\to v;j}\phi_j(\cdot)\right]_+$, for all $w \in \mathcal{N}(v) \setminus \{u\}$.

   (b) Draw k i.i.d. samples Y_i^{t+1} from the probability density proportional to

$$\beta_{v\to u}(y) \prod_{w\in\mathcal{N}(v)\setminus\{u\}} \widehat{m}^t_{w\to v}(y),$$

   where $\beta_{v\to u}$ was previously defined in Equation (8).

   (c) Use the samples $\{Y_1^{t+1},\ldots,Y_k^{t+1}\}$ from step (b) to compute

$$\widetilde{b}^{t+1}_{v\to u;j} := \frac{1}{k}\sum_{i=1}^k \gamma_{v\to u;j}(Y_i^{t+1}) \qquad \text{for } j = 1,2,\ldots,r, \tag{12}$$

   where the function $\gamma_{v\to u;j}$ is defined in Equation (10).

   (d) For step size $\eta^t = 1/(t+1)$, update the r-dimensional message coefficient vectors $a^t_{v\to u} \mapsto a^{t+1}_{v\to u}$ via

$$a^{t+1}_{v\to u} = (1-\eta^t)\, a^t_{v\to u} + \eta^t\, \widetilde{b}^{t+1}_{v\to u}. \tag{13}$$

Figure 3: The *SOSMP* algorithm for continuous state space marginalization.

At each iteration $t = 0, 1, 2, \ldots$, the algorithm maintains an r-dimensional vector of basis expansion coefficients

$$a^t_{v\to u} = (a^t_{v\to u;1},\ldots,a^t_{v\to u;r}) \in \mathbb{R}^r, \quad \text{on directed edge } (v \to u) \in \vec{\mathcal{E}}.$$

This vector should be understood as defining the current message approximation $m^t_{v\to u}$ on edge $(v \to u)$ via the expansion

$$m^t_{v\to u}(\cdot) := \sum_{j=1}^r a^t_{v\to u;j}\,\phi_j(\cdot). \tag{11}$$

We use $a^t = \left\{a^t_{v\to u}, (v \to u) \in \vec{\mathcal{E}}\right\}$ to denote the full set of $r|\vec{\mathcal{E}}|$ coefficients that are maintained by the algorithm at iteration t. With this notation, the algorithm consists of a sequence of steps, detailed in Figure 3, that perform the update $a^t \mapsto a^{t+1}$, and hence implicitly the update $m^t \mapsto m^{t+1}$.

3.4 Computational Complexity

The sampling in Step 2(b) of the algorithm may be performed using standard sampling methods (Ripley, 1987). For instance, accept-reject sampling involves drawing from an auxiliary distribution, and then performing an accept-reject step. Let c denote the average number of times that an auxiliary sample must be drawn until acceptance. From Figure 3, it can be seen that each iteration requires $O(crk)$ floating point operations on average per edge. Note that the accept-reject factor c depends only on the state space $\mathcal{X}$ and the auxiliary distribution, and it is independent of r, k, and the graph structure.

It is also interesting to compare the computational complexity of our algorithm to that of competing methods. For instance, the non-parametric (Gaussian mixture) BP algorithm, as proposed in past works (Sudderth et al., 2010; Isard, 2003), involves approximating each message with an ℓ-component mixture of Gaussians. The associated computational complexity is $O(\ell^{d_{\max}})$ per iteration per edge, where $d_{\max}$ is the maximum degree of the graph, so that the method is suitable for graphs of relatively low degree. Ihler and McAllester (2009) suggested an improvement known as particle BP, in which messages are approximated by particles associated with the nodes (as opposed to the edges). Given an approximation based on ℓ particles at each node, the computational complexity of particle BP is $O(\ell^2)$ per iteration per edge. More recently, Song et al. (2011) proposed a method known as kernel BP, designed for jointly estimating the potential functions and computing (approximate) marginal distributions. Their approach is based on representing the potentials as weighted sums of kernels in a reproducing kernel Hilbert space, which leads to simple BP updates. For a training set with M samples, the computational complexity of the kernel BP is $O(M^2)$ per iteration per edge. They also proposed a more efficient variant of the kernel BP by approximating the feature matrices by weighted combination of subsets of their columns.

4. Theoretical Guarantees

We now turn to the theoretical analysis of the SOSMP algorithm, and guarantees relative to the fixed points of the true BP algorithm. For any tree-structured graph, the BP algorithm is guaranteed to have a unique message fixed point $m^* = \{m^*_{v\to u},\ (v \to u) \in \vec{\mathcal{E}}\}$. For graphs with cycles, uniqueness is no longer guaranteed, which would make it difficult to compare with the SOSMP algorithm. Accordingly, in our analysis of the loopy graph, we make a natural contractivity assumption, which guarantees uniqueness of the fixed point m^*.

The SOSMP algorithm generates a random sequence $\{a^t\}_{t=0}^{\infty}$, which define message approximations $\{m^t\}_{t=0}^{\infty}$ via the expansion (11). Of interest to us are the following questions:

- under what conditions do the message iterates approach a neighborhood of the BP fixed point m^* as $t \to +\infty$?
- when such convergence takes place, how fast is it?

In order to address these questions, we separate the error in our analysis into two terms: algorithmic error and approximation error. For a given r, let Π^r denote the projection operator onto the span of $\{\phi_1, \ldots, \phi_r\}$. In detail, given a function f represented in terms of the infinite series

expansion $f = \sum_{j=1}^{\infty} a_j \phi_j$, we have

$$\Pi^r(f) \;:=\; \sum_{j=1}^{r} a_j \phi_j.$$

For each directed edge $(v \to u) \in \vec{\mathcal{E}}$, define the functional error

$$\Delta^t_{v\to u} \;:=\; m^t_{v\to u} - \Pi^r(m^*_{v\to u}) \tag{14}$$

between the message approximation at time t, and the BP fixed point projected onto the first r basis functions. Moreover, define the approximation error at the BP fixed point as

$$A^r_{v\to u} \;:=\; m^*_{v\to u} - \Pi^r(m^*_{v\to u}). \tag{15}$$

Since $\Delta^t_{v\to u}$ belongs to the span of the first r basis functions, the Pythagorean theorem implies that the overall error can be decomposed as

$$\|m^t_{v\to u} - m^*_{v\to u}\|^2_{L^2} \quad = \quad \underbrace{\|\Delta^t_{v\to u}\|^2_{L^2}}_{\text{Estimation error}} \quad + \quad \underbrace{\|A^r_{v\to u}\|^2_{L^2}}_{\text{Approximation error}} .$$

Note that the approximation error term is independent of the iteration number t, and can only be reduced by increasing the number r of coefficients used in the series expansion. Our analysis of the estimation error is based on controlling the $|\vec{\mathcal{E}}|$-dimensional error vector

$$\rho^2\big(\Delta^t\big) \;:=\; \big\{\|\Delta^t_{v\to u}\|^2_{L^2},\, (v\to u) \in \vec{\mathcal{E}}\big\} \;\in\; \mathbb{R}^{|\vec{\mathcal{E}}|}, \tag{16}$$

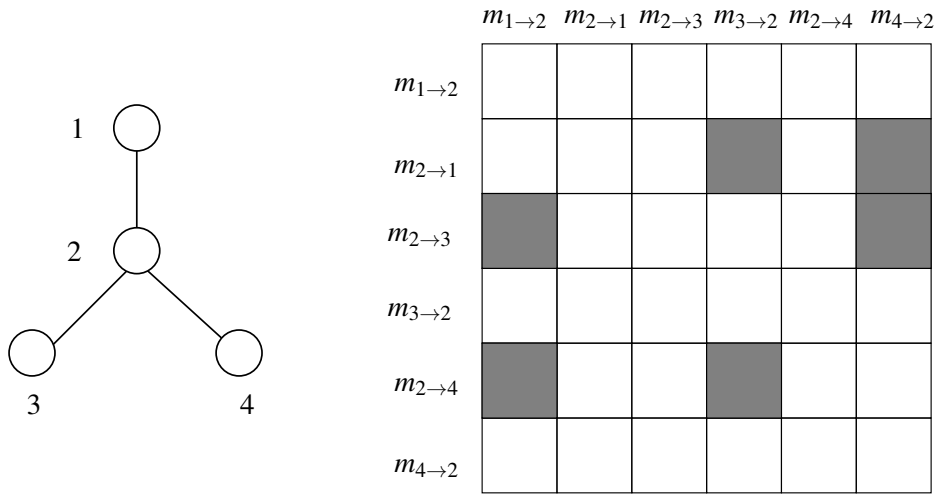
and in particular showing that it decreases as $O(1/t)$ up to an error floor imposed by the approximation error. In order to analyze the error, we also introduce the $|\vec{\mathcal{E}}|$-dimensional vector of approximation errors

$$\rho^2\big(A^r\big) := \big\{\|A^r_{v\to u}\|^2_{L^2},\, (v\to u) \in \vec{\mathcal{E}}\big\} \in \mathbb{R}^{|\vec{\mathcal{E}}|}. \tag{17}$$

By increasing r, we can reduce this approximation error term, but at the same time, we increase the computational complexity of each update. In Section 4.3, we discuss how to choose r so as to trade-off the estimation and approximation errors with computational complexity.

4.1 Bounds for Tree-Structured Graphs

With this set-up, we now turn to bounds for tree-structured graphs. Our analysis of the tree-structured case controls the vector of estimation errors $\rho^2\big(\Delta^t\big)$ using a nilpotent matrix $N \in \mathbb{R}^{|\vec{\mathcal{E}}|\times|\vec{\mathcal{E}}|}$ determined by the tree structure (Noorshams and Wainwright, 2013). Recall that a matrix N is nilpotent with order ℓ if $N^\ell = 0$ and $N^{\ell-1} \neq 0$ for some ℓ. As illustrated in Figure 4, the rows and columns of N are indexed by directed edges. For the row indexed by $(v \to u)$, there can be non-zero entries only for edges in the set $\{(w \to v),\, w \in \mathcal{N}(v)\backslash\{u\}\}$. These directed edges are precisely those that pass messages relevant in updating the message from v to u, so that N tracks the propagation of message information in the graph. As shown in our previous work (see Lemma 1 in the paper by Noorshams and Wainwright, 2013), the matrix N with such structure is nilpotent with degree at



(a) (b)

Figure 4: (a) A simple tree with $|\mathcal{E}| = 3$ edges and hence $|\vec{\mathcal{E}}| = 6$ directed edges. (b) Structure of nilpotent matrix $N \in \mathbb{R}^{|\vec{\mathcal{E}}|\times|\vec{\mathcal{E}}|}$ defined by the graph in (a). Rows and columns of the matrix are indexed by directed edges $(v \to u) \in \vec{\mathcal{E}}$; for the row indexed by $(v \to u)$, there can be non-zero entries only for edges in the set $\{(w \to v),\, w \in \mathcal{N}(v)\backslash\{u\}\}$.

most the diameter of the tree. (In a tree, there is always a unique edge-disjoint path between any pair of nodes; the diameter of the tree is the length of the longest of these paths.)

Moreover, our results on tree-structured graphs impose one condition on the vector of approximation errors A^r, namely that

$$\inf_{y\in\mathcal{X}} \Pi^r\big(\Gamma_{uv}(x,y)\big) \;>\; 0, \quad \text{and} \quad |A^r_{v\to u}(x)| \;\leq\; \frac{1}{2} \inf_{y\in\mathcal{X}} \Pi^r\big(\Gamma_{uv}(x,y)\big) \tag{18}$$

for all $x \in \mathcal{X}$ and all directed edges $(v \to u) \in \vec{\mathcal{E}}$. This condition ensures that the L^2-norm of the approximation error is not too large relative to the compatibility functions. Since $\sup_{x,y\in\mathcal{X}} |\Pi^r\big(\Gamma_{uv}(x,y)\big) - \Gamma_{uv}(x,y)| \to 0$ and $\sup_{x\in\mathcal{X}} |A^r_{v\to u}(x)| \to 0$ as $r \to +\infty$, assuming that the compatibility functions are uniformly bounded away from zero, condition (18) will hold once the number of basis expansion coefficients r is sufficiently large. Finally, our bounds involve the constants

$$B_j \;:=\; \max_{(v\to u)\in\vec{\mathcal{E}}} \sup_{y\in\mathcal{X}} \langle \Gamma_{uv}(\cdot, y), \phi_j \rangle_{L^2}. \tag{19}$$

With this set-up, we have the following guarantees:

Theorem 2 (tree-structured graphs) *Suppose that $\mathcal{X}$ is closed and bounded, the node and edge potential functions are continuous, and that condition* (18) *holds. Then for any tree-structured model, the sequence of messages $\{m^t\}_{t=0}^{\infty}$ generated by the SOSMP algorithm have the following properties:*

(a) There is a nilpotent matrix $N \in \mathbb{R}^{|\vec{\mathcal{E}}|\times|\vec{\mathcal{E}}|}$ *such that the error vector* $\rho^2(\Delta^t)$ *converges almost surely to the set*

$$\mathcal{B} := \{e \in \mathbb{R}^{|\vec{\mathcal{E}}|} \mid |e| \preceq N(I-N)^{-1}\rho^2(A^r)\},$$

where $\preceq$ *denotes elementwise inequality for vectors.*

(b) Furthermore, for all iterations $t = 1,2,\ldots$*, we have*

$$\mathbb{E}[\rho^2(\Delta^t)] \preceq \left(12\sum_{j=1}^{r} B_j^2\right)\frac{(I-\log t\, N)^{-1}}{t}(N\vec{1}+8) + N(I-N)^{-1}\rho^2(A^r).$$

To clarify the statement in part (a), it guarantees that the difference $\rho^2(\Delta^t) - \Pi_{\mathcal{B}}(\rho^2(\Delta^t))$ between the error vector and its projection onto the set $\mathcal{B}$ converges almost surely to zero. Part (b) provides a quantitative guarantee on how quickly the expected absolute value of this difference converges to zero. In particular, apart from logarithmic factors in t, the convergence rate guarantees is of the order $O(1/t)$.

4.2 Bounds for General Graphs

Our next theorem addresses the case of general graphical models. The behavior of the ordinary BP algorithm to a graph with cycles—in contrast to the tree-structured case—is more complicated. On one hand, for strictly positive potential functions (as considered in this paper), a version of Brouwer's fixed point theorem can be used to establish existence of fixed points (Wainwright and Jordan, 2008). However, in general, there may be multiple fixed points, and convergence is not guaranteed. Accordingly, various researchers have studied conditions that are sufficient to guarantee uniqueness of fixed points and/or convergence of the ordinary BP algorithm: one set of sufficient conditions, for both uniqueness and convergence, involve assuming that the BP update operator is a contraction in a suitable norm (e.g., Tatikonda and Jordan, 2002; Ihler et al., 2005; Mooij and Kappen, 2007; Roosta et al., 2008).

In our analysis of the SOSMP algorithm for a general graph, we impose the following form of *contractivity*: there exists a constant $0 < \gamma < 2$ such that

$$\|\mathcal{F}_{v\to u}(m) - \mathcal{F}_{v\to u}(m')\|_{L^2} \leq \left(1-\frac{\gamma}{2}\right)\sqrt{\frac{1}{|\mathcal{N}(v)\backslash\{u\}|}\sum_{w\in\mathcal{N}(v)\backslash\{u\}}\|m_{w\to v} - m'_{w\to v}\|_{L^2}^2}, \tag{20}$$

for all directed edges $(v \to u) \in \vec{\mathcal{E}}$, and feasible messages m, and m'. We say that the ordinary BP algorithm is γ-contractive when condition (20) holds.

Theorem 3 (general graphs) *Suppose that the ordinary BP algorithm is* γ*-contractive* (20)*, and consider the sequence of messages* $\{m^t\}_{t=0}^{\infty}$ *generated with step-size* $\eta^t = 1/(\gamma(t+1))$*. Then for all* $t = 1,2,\ldots$*, the error sequence* $\{\Delta_{v\to u}^t\}_{t=0}^{\infty}$ *is bounded in mean-square as*

$$\mathbb{E}[\rho^2(\Delta^t)] \preceq \left[\left(\frac{8\sum_{j=1}^r B_j^2}{\gamma^2}\right)\frac{\log t}{t} + \frac{1}{\gamma}\max_{(v\to u)\in\vec{\mathcal{E}}}\|A_{v\to u}^r\|_{L^2}^2\right]\vec{1}. \tag{21}$$

where $A_{v\to u}^r = m_{v\to u}^* - \Pi^r(m_{v\to u}^*)$ *is the approximation error on edge* $(v \to u)$*.*

Theorem 3 guarantees that under the contractivity condition (20), the SOSMP iterates converge to a neighborhood of the BP fixed point. The error offset depends on the approximation error term that decays to zero as r is increased. Moreover, disregarding the logarithmic factor, the convergence rate is $O(1/t)$, which is the best possible for a stochastic approximation scheme of this type (Nemirovsky and Yudin, 1983; Agarwal et al., 2012).

4.3 Explicit Rates for Kernel Classes

Theorems 2 and 3 are generic results that apply to any choices of the edge potential functions. In this section, we pursue a more refined analysis of the number of arithmetic operations that are required to compute a δ-*uniformly accurate* approximation to the BP fixed point m^*, where $\delta > 0$ is a user-specified tolerance parameter. By a δ-uniformly accurate approximation, we mean a collection of messages m such that

$$\max_{(v\to u)\in\vec{\mathcal{E}}} \mathbb{E}\big[\|m_{v\to u} - m^*_{v\to u}\|_{L^2}^2\big] \;\leq\; \delta.$$

In order to obtain such an approximation, we need to specify both the number of coefficients r to be retained, and the number of iterations that we should perform. Based on these quantities, our goal is to specify the *minimal number of basic arithmetic operations* $T(\delta)$ that are sufficient to compute a δ-accurate message approximation.

In order to obtain concrete answers, we study this issue in the context of kernel-based potential functions. In many applications, the edge potentials $\psi_{uv} : \mathcal{X}\times\mathcal{X} \to \mathbb{R}_+$ are symmetric and positive semidefinite (PSD) functions, frequently referred to as kernel functions.[2] Commonly used examples include the Gaussian kernel $\psi_{uv}(x,y) = \exp(-\gamma\|x-y\|_2^2)$, the closely related Laplacian kernel, and other types of kernels that enforce smoothness priors. Any kernel function defines a positive semidefinite integral operator, namely via Equation (7). When $\mathcal{X}$ is compact and the kernel function is continuous, then Mercer's theorem (Riesz and Nagy, 1990) guarantees that this integral operator has a countable set of eigenfunctions $\{\phi_j\}_{j=1}^{\infty}$ that form an orthonormal basis of $L^2(\mathcal{X};\mu)$. Moreover, the kernel function has the expansion

$$\psi_{uv}(x,y) \;=\; \sum_{j=1}^{\infty} \lambda_j\,\phi_j(x)\,\phi_j(y),$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ are the eigenvalues, all guaranteed to be non-negative. In general, the eigenvalues might differ from edge to edge, but we suppress this dependence for simplicity in exposition. We study kernels that are trace class, meaning that the eigenvalues are absolutely summable (i.e., $\sum_{j=1}^{\infty}\lambda_j < \infty$).

For a given eigenvalue sequence $\{\lambda_j\}_{j=1}^{\infty}$ and some tolerance $\delta > 0$, we define the *critical dimension* $r^* = r^*(\delta;\{\lambda_j\})$ to be the smallest positive integer r such that

$$\lambda_r \leq \delta. \tag{22}$$

Since $\lambda_j \to 0$, the existence of $r^* < \infty$ is guaranteed for any $\delta > 0$.

2. In detail, a PSD kernel function has the property that for all natural numbers n and $\{x_1,\ldots,x_n\} \subset \mathcal{X}$, the $n\times n$ kernel matrix with entries $\psi_{uv}(x_i,x_j)$ is symmetric and positive semidefinite.

Theorem 4 (complexity vs. accuracy) *In addition to the conditions of Theorem 3, suppose that the compatibility functions are defined by a symmetric PSD trace-class kernel with eigenvalues $\{\lambda_j\}$. If we run the SOSMP algorithm with $r^* = r^*(\delta;\{\lambda_j\})$ basis coefficients, and $k = O(1)$ samples, then it suffices to perform*

$$T(\delta;\{\lambda_j\}) \;=\; O\Big(r^*\,\big(\sum_{j=1}^{r^*}\lambda_j^2\big)\,\big(1/\delta\big)\,\log(1/\delta)\Big) \tag{23}$$

arithmetic operations per edge in order to obtain a δ-accurate message vector m.

The proof of Theorem 4 is provided in Section 6.3. It is based on showing that the choice (22) suffices to reduce the approximation error to $O(\delta)$, and then bounding the total operation complexity required to also reduce the estimation error.

Theorem 4 can be used to derive explicit estimates of the complexity for various types of kernel classes. We begin with the case of kernels in which the eigenvalues decay at an inverse polynomial rate: in particular, given some $\alpha > 1$, we say that they exhibit α-*polynomial decay* if there is a universal constant C such that

$$\lambda_j \leq C/j^{\alpha} \quad \text{for all } j = 1,2,\ldots. \tag{24}$$

Examples of kernels in this class include Sobolov spline kernels (Gu, 2002), which are a widely used type of smoothness prior. For example, the spline class associated with functions that are s-times differentiable satisfies the decay condition (24) with $\alpha = 2s$.

Corollary 5 *In addition to the conditions of Theorem 3, suppose that the compatibility functions are symmetric kernels with α-polynomial decay* (24)*. Then it suffices to perform*

$$T_{\text{poly}}(\delta) = O\Big(\big(1/\delta\big)^{\frac{1+\alpha}{\alpha}}\log(1/\delta)\Big)$$

operations per edge in order to obtain a δ-accurate message vector m.

The proof of this corollary is immediate from Theorem 4: given the assumption $\lambda_j \leq C/j^{\alpha}$, we see that $r^* \leq (C/\delta)^{\frac{1}{\alpha}}$ and $\sum_{j=1}^{r^*}\lambda_j^2 = O(1)$. Substituting into the bound (23) yields the claim. Corollary 5 confirms a natural intuition—namely, that it should be easier to compute an approximate BP fixed point for a graphical model with smooth potential functions. Disregarding the logarithmic factor (which is of lower-order), the operation complexity $T_{\text{poly}}(\delta)$ ranges from $O\big((1/\delta)^2\big)$, obtained as $\alpha \to 1^+$ all the way down to $O\big(1/\delta\big)$, obtained as $\alpha \to +\infty$.

Another class of widely used kernels are those with exponentially decaying eigenvalues: in particular, for some $\alpha > 0$, we say that the kernel has α-*exponential decay* if there are universal constants (C,c) such that

$$\lambda_j \leq C\exp(-cj^{\alpha}) \qquad \text{for all } j = 1,2,\ldots. \tag{25}$$

Examples of such kernels include the Gaussian kernel, which satisfies the decay condition (25) with $\alpha = 2$ (e.g., Steinwart and Christmann, 2008).

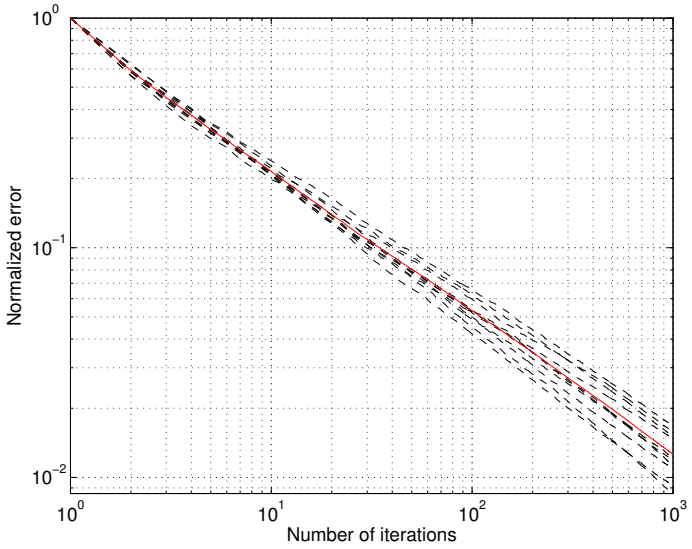


Figure 5: Plot of normalized error e^t/e^0 vs. the number of iterations t for 10 different sample paths on a chain of size $n = 100$. The dashed lines are sample paths whereas the solid line is the mean square error. In this experiment node and edge potentials are mixtures of three Gaussians and we implemented SOSMP using the first $r = 10$ Fourier coefficients with $k = 5$ samples.

Corollary 6 *In addition to the conditions of Theorem 3, suppose that the compatibility functions are symmetric kernels with* α*-exponential decay* (25)*. Then it suffices to perform*

$$T_{\exp}(\delta) = O\Big((1/\delta)\left(\log(1/\delta)\right)^{\frac{1+\alpha}{\alpha}}\Big) \tag{26}$$

operations per edge in order to obtain a uniformly δ*-accurate message vector m.*

As with our earlier corollary, the proof of this claim is a straightforward consequence of Theorem 4. Corollary 6 demonstrates that kernel classes with exponentially decaying eigenvalues are not significantly different from parametric function classes, for which a stochastic algorithm would have operation complexity $O(1/\delta)$. Apart from the lower order logarithmic terms, the complexity bound (26) matches this parametric rate.

5. Experimental Results

In this section, we describe some experimental results that help to illustrate the theoretical predictions of the previous section.

5.1 Synthetic Data

We begin by running some experiments for a simple model, in which both the node and edge potentials are mixtures of Gaussians. More specifically, we form a graphical model with potential

functions of the form

$$\psi_u(x_u) = \sum_{i=1}^{3} \pi_{u;i} \exp\big(-(x_u - \mu_{u;i})^2/(2\sigma_{u;i}^2)\big), \quad \text{for all } u \in \mathcal{V}, \text{ and} \tag{27}$$

$$\psi_{uv}(x_u, x_v) = \sum_{i=1}^{3} \pi_{uv;i} \exp\big(-(x_v - x_u)^2/(2\sigma_{uv;i}^2)\big) \quad \text{for all } (u,v) \in \mathcal{E}, \tag{28}$$

where the non-negative mixture weights are normalized (i.e., $\sum_{i=1}^{3} \pi_{uv;i} = \sum_{i=1}^{3} \pi_{u;i} = 1$). For each vertex and edge and for all $i = 1,2,3$, the mixture parameters are chosen randomly from uniform distributions over the range $\sigma_{u;i}^2, \sigma_{uv;i}^2 \in (0, 0.5]$ and $\mu_{u;i} \in [-3,3]$.

5.1.1 SAMPLE PATHS ON A TREE-STRUCTURED GRAPH

For a chain-structured graph with $n = 100$ nodes, we first compute the fixed point of the standard BP, using direct numerical integration to compute the integrals,[3] so to compute (a very accurate approximation of) the fixed point m^*. We compare this "exact" answer to the approximation obtained by running the SOSMP algorithm using the first $r = 10$ Fourier basis coefficients and $k = 5$ samples. Having run the SOSMP algorithm, we compute the average squared error

$$e^t := \frac{1}{|\vec{\mathcal{E}}|} \sum_{(v \to u) \in \vec{\mathcal{E}}} \sum_{j=1}^{r} (a_{v \to u;j}^t - a_{v \to u;j}^*)^2 \tag{29}$$

at each time $t = 1, 2, \ldots$.

Figure 5 provides plots of the error (29) versus the number of iterations for 10 different trials of the SOSMP algorithm. (Since the algorithm is randomized, each path is slightly different.) The plots support our claim of of almost sure convergence, and moreover, the straight lines seen in the log-log plots confirm that convergence takes place at a rate inverse polynomial in t.

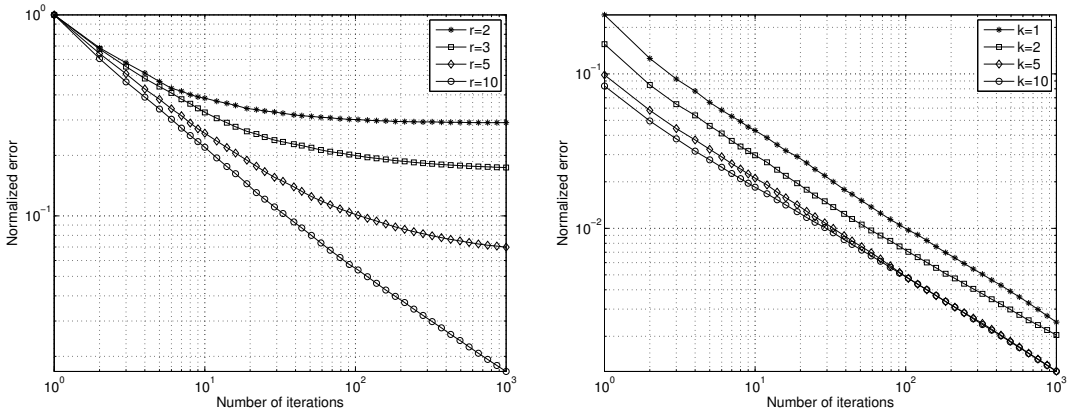
5.1.2 EFFECT OF THE NUMBER OF COEFFICIENTS r AND THE NUMBER OF SAMPLES k

In the next few simulations, we test the algorithm's behavior with respect to the number of expansion coefficients r, and number of samples k. In particular, Figure 6(a) illustrates the expected error, averaged over several sample paths, vs. the number of iterations for different number of expansion coefficients $r \in \{2,3,5,10\}$ when $k = 5$ fixed; whereas Figure 6(b) depicts the expected error vs. the number of iterations for different number of samples $k \in \{1,2,5,10\}$ when $r = 10$ is fixed. As expected, in Figure 6(a), the error decreases monotonically, with the rate of $1/t$, till it hits a floor corresponding the offset incurred by the approximation error. Moreover, the error floor decreases with the number of expansion coefficients. On the other hand, in Figure 6(b), increasing the number of samples causes a downward shift in the error. This behavior is also expected since increasing the number of samples reduces the variance of the empirical expectation in Equation (12).

5.1.3 EFFECT OF THE EDGE POTENTIAL SMOOTHNESS

In our next set of experiments, still on a chain with $n = 100$ vertices, we test the behavior of the SOSMP algorithm on graphs with edge potentials of varying degrees of smoothness. In all cases,

3. In particular, we approximate the integral update (4) with its Riemann sum over the range $\mathcal{X} = [-5,5]$ and with 100 samples per unit time.



(a) (b)

Figure 6: Normalized mean squared error $\mathbb{E}[e^t/e^0]$ verses the number of iterations for a Markov chain with $n = 100$ nodes, using potential functions specified by the mixture of Gaussians model (27) and (28). (a) Behavior as the number of expansion coefficients is varied over the range $r \in \{2,3,5,10\}$ with $k = 5$ samples in all cases. As predicted by the theory, the error drops monotonically with the number of iterations until it hits a floor. The error floor, which corresponds to the approximation error incurred by message expansion truncation, decreases as the number of coefficients r is increased. (b) Mean squared error $\mathbb{E}[e^t]$ verses the number of iterations t for different number of samples $k \in \{1,2,5,10\}$, in all cases using $r = 10$ coefficients. Increasing the number of samples k results in a downward shift in the error.

we use node potentials from the Gaussian mixture ensemble (27) previously discussed, but form the edge potentials in terms of a family of kernel functions. More specifically, consider the basis functions

$$\phi_j(x) = \sin\big((2j-1)\pi(x+5)/10\big) \quad \text{for } j = 1,2,\ldots.$$

each defined on the interval $[-5,5]$. It is straightforward that the family $\{\phi_j\}_{j=1}^{\infty}$ forms an orthonormal basis of $L^2[-5,5]$. We use this basis to form the edge potential functions

$$\psi_{uv}(x,y) = \sum_{j=1}^{1000} (1/j)^{\alpha} \phi_j(x)\, \phi_j(y),$$

where $\alpha > 0$ is a parameter to be specified. By construction, each edge potential is a positive semidefinite kernel function satisfying the α-polynomial decay condition (24).

Figure 7 illustrate the error curves for two different choices of the smoothness parameter: panel (a) shows $\alpha = 0.1$, whereas panel (b) shows $\alpha = 1$. For the larger value of α shown in panel (b), the messages in the BP algorithm are smoother, so that the SOSMP estimates are more accurate with the same number of expansion coefficients. Moreover, similar to what we have observed previously, the error decays with the rate of $1/t$ till it hits the error floor. Note that this error floor is lower for

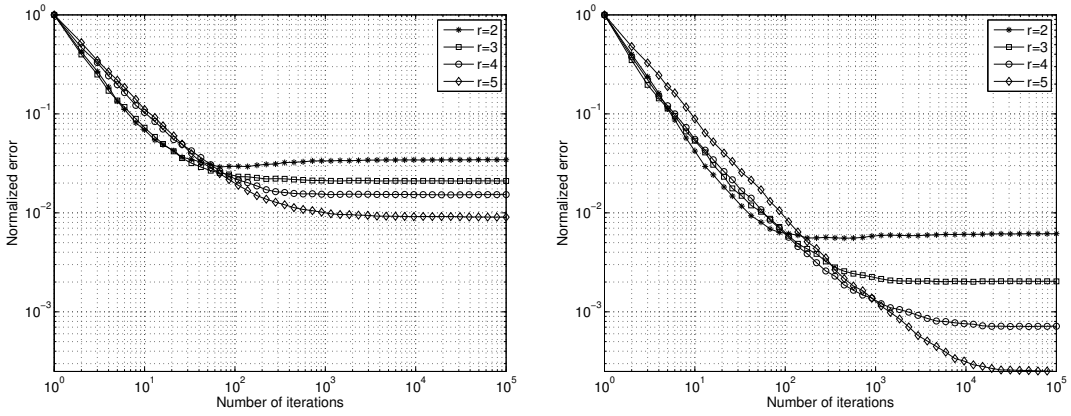


Figure 7: Plot of the estimation error e^t/e^0 verses the number of iterations t for the cases of (a) $\alpha = 0.1$ and (b) $\alpha = 1$. The BP messages are smoother when $\alpha = 1$, and accordingly the SOSMP estimates are more accurate with the same number of expansion coefficients. Moreover, the error decays with the rate of $1/t$ till it hits a floor corresponding to the approximation error incurred by truncating the message expansion coefficients.

the smoother kernel ($\alpha = 1$) compared to the rougher case ($\alpha = 0.1$). Moreover, as predicted by our theory, the approximation error decays faster for the smoother kernel, as shown by the plots in Figure 8, in which we plot the final error, due purely to approximation effects, versus the number of expansion coefficients r. The semilog plot of Figure 8 shows that the resulting lines have different slopes, as would be expected.

5.2 Computer Vision Application

Moving beyond simulated problems, we conclude by showing the SOSMP algorithm in application to a larger scale problem that arises in computer vision—namely, that of optical flow estimation (Baker et al., 2011). In this problem, the input data are two successive frames of a video sequence. We model each frame as a collection of pixels arrayed over a $\sqrt{n} \times \sqrt{n}$ grid, and measured intensity values at each pixel location of the form $\{I(i,j), I'(i,j)\}_{i,j=1}^{\sqrt{n}}$. Our goal is to estimate a 2-dimensional motion vector $x_u = (x_{u;1}, x_{u;2})$ that captures the local motion at each pixel $u = (i,j)$, $i,j = 1,2,\ldots,\sqrt{n}$ of the image sequence.

In order to cast this optical flow problem in terms of message-passing on a graph, we adopt the model used by Boccignone et al. (2007). We model the local motion X_u as a 2-dimensional random vector taking values in the space $\mathcal{X} = [-d,d] \times [-d,d]$, and associate the random vector X_u with vertex u, in a 2-dimensional grid (see Figure 1(a)). At node $u = (i,j)$, we use the change between the two image frames to specify the node potential

$$\psi_u(x_{u;1}, x_{u;2}) \propto \exp\left(- \frac{(I(i,j) - I'(i + x_{u;1},\ j + x_{u;2}))^2}{2\sigma_u^2} \right).$$

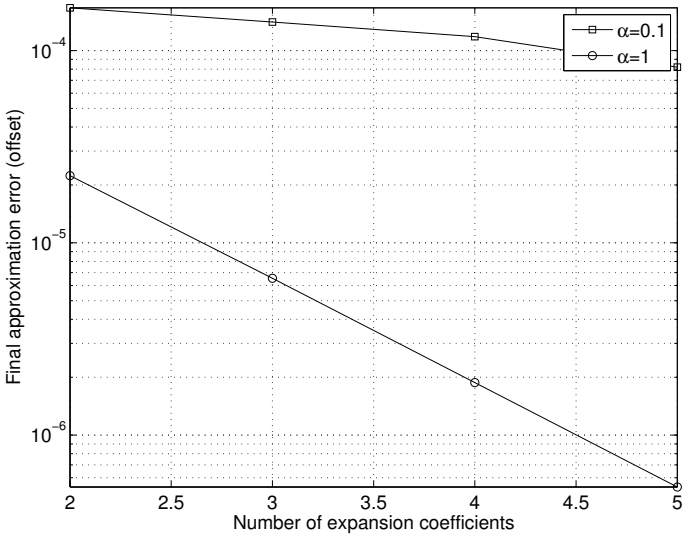


Figure 8: Final approximation error vs. the number of expansion coefficients for the cases of $\alpha = 0.1$ and $\alpha = 1$. As predicted by the theory, the error floor decays with a faster pace for the smoother edge potential.

(a) (b)

Figure 9: Two frames, each of dimension 250×250 pixels, taken from a video sequence of moving cars.

On each edge (u,v), we introduce the potential function

$$\psi_{uv}(x_u, x_v) \propto \exp\left(-\frac{\|x_u - x_v\|^2}{2\sigma_{uv}^2}\right),$$

which enforces a type of smoothness prior over the image.

To estimate the motion of a truck, we applied the SOSMP algorithm using the 2-dimensional Fourier expansion as our orthonormal basis to two 250×250 frames from a truck video sequence (see Figure 9). We apply the SOSMP algorithm using the first $r = 9$ coefficients and $k = 3$ samples. Figure 10 shows the HSV (hue, saturation, value) codings of the estimated motions after $t = 1, 10, 40$ iterations, in panels (a), (b) and (c) respectively. Panel (d) provides an illustration of the HSV encoding: hue is used to represent the angular direction of the motion whereas the speed (magnitude of the motion) is encoded by the saturation (darker colors meaning higher speeds). The initial estimates of the motion vectors are noisy, but it fairly rapidly converges to a reasonable optical flow field. (To be clear, the purpose of this experiment is not to show the effectiveness of SOSMP or BP as a particular method for optical flow, but rather to demonstrate its correctness and feasibility of the SOSMP in an applied setting.)

6. Proofs

We now turn to the proofs of our main results, which involve various techniques from concentration of measure, stochastic approximation, and functional analysis. For the reader's convenience, we provide a high-level outline here. We begin in Section 6.1 by proving Theorem 2 dealing with the case of tree-structured graphical models. In Section 6.2, we turn to the proof of Theorem 3 concerning the case of general graphs. Finally, in Section 6.3 we characterize the trade-off between computational complexity and accuracy by proving Theorem 4. Each section involve technical steps some of which are presented as lemmas. To increase the readability of the paper, proofs of the lemmas are deferred to the appendices and can be ignored without affecting the flow of the main argument.

6.1 Trees: Proof of Theorem 2

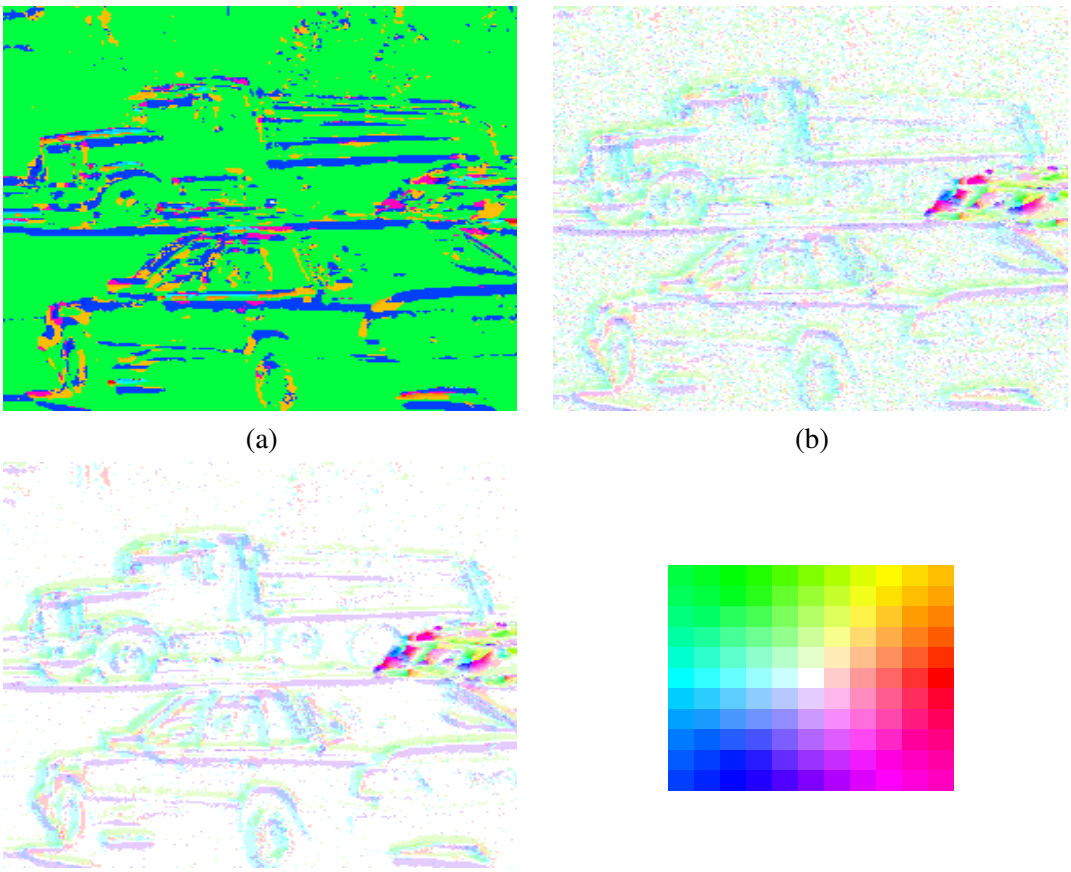
Our goal is to bound the error

$$\|\Delta_{v\to u}^{t+1}\|_{L^2}^2 \;=\; \|m_{v\to u}^{t+1} - \Pi^r(m_{v\to u}^*)\|_{L^2}^2 \;=\; \sum_{j=1}^{r} \big(a_{v\to u;j}^{t+1} - a_{v\to u;j}^*\big)^2, \tag{30}$$

where the final equality follows by Parseval's theorem. Here $\{a_{v\to u;j}^*\}_{j=1}^r$ are the basis expansion coefficients that define the best r-approximation to the BP fixed point m^*. The following lemma provides an upper bound on this error in terms of two related quantities. First, we let $\{b_{v\to u;j}^t\}_{j=1}^\infty$ denote the basis function expansion coefficients of the $\mathcal{F}_{v\to u}(\widehat{m}^t)$—that is, $[\mathcal{F}_{v\to u}(\widehat{m}^t)](\cdot) = \sum_{j=1}^\infty b_{v\to u;j}^t \phi_j(\cdot)$. Second, for each $j = 1, 2, \ldots, r$, define the deviation $\zeta_{v\to u;j}^{t+1} := \widetilde{b}_{v\to u;j}^{t+1} - b_{v\to u;j}^t$, where the coefficients $\widetilde{b}_{v\to u;j}^{t+1}$ are updated in Step 2(c) Figure 3.

Lemma 7 *For each iteration $t = 0, 1, 2, \ldots$, we have*

$$\|\Delta_{v\to u}^{t+1}\|_{L^2}^2 \;\leq\; \underbrace{\frac{2}{t+1}\sum_{j=1}^{r}\sum_{\tau=0}^{t}\big[b_{v\to u;j}^{\tau} - a_{v\to u;j}^*\big]^2}_{\text{Deterministic term } D_{v\to u}^{t+1}} + \underbrace{\frac{2}{(t+1)^2}\sum_{j=1}^{r}\Big[\sum_{\tau=0}^{t}\zeta_{v\to u;j}^{\tau+1}\Big]^2}_{\text{Stochastic term } S_{v\to u}^{t+1}}. \tag{31}$$



(a) (b)

(c) (d)

Figure 10: Color coded images of the estimated motion vectors after (a) $t = 1$, (b) $t = 10$, (c) $t = 40$ iterations. Panel (d) illustrates the hsv color coding of the flow. The color hue is used to encode the angular dimension of the motion, whereas the saturation level corresponds to the speed (length of motion vector). We implemented the SOSMP algorithm by expanding in the two-dimensional Fourier basis, using $r = 9$ coefficients and $k = 3$ samples. Although the initial estimates are noisy, it converges to a reasonable optical flow estimate after around 40 iterations.

The proof of this lemma is relatively straightforward; see Appendix A for the details. Note that inequality (31) provides an upper bound on the error that involves two terms: the first term $D^{t+1}_{v\to u}$ depends only on the expansion coefficients $\{b^{\tau}_{v\to u;j}, \tau = 0,\ldots,t\}$ and the BP fixed point, and therefore is a deterministic quantity when we condition on all randomness in stages up to step t. The second term $S^{t+1}_{v\to u}$, even when conditioned on randomness through step t, remains stochastic, since the coefficients $b^{t+1}_{v\to u}$ (involved in the error term $\zeta^{t+1}_{v\to u}$) are updated stochastically in moving from iteration t to $t+1$.

We split the remainder of our analysis into three parts: (a) control of the deterministic component; (b) control of the stochastic term; and (c) combining the pieces to provide a convergence bound.

6.1.1 UPPER-BOUNDING THE DETERMINISTIC TERM

By the Pythagorean theorem, we have

$$\sum_{\tau=0}^{t}\sum_{j=1}^{r}\left[b^{\tau}_{v\to u;j}-a^{*}_{v\to u;j}\right]^2 \;\leq\; \sum_{\tau=0}^{t}\|\mathcal{F}_{v\to u}(\widehat{m}^t)-\mathcal{F}_{v\to u}(m^*)\|_{L^2}^2. \tag{32}$$

In order to control this term, we make use of the following lemma, proved in Appendix B:

Lemma 8 *For all directed edges $(v\to u)\in\vec{E}$, there exist constants $\{L_{v\to u;w}, w\in\mathcal{N}(v)\setminus\{u\}\}$ such that*

$$\|\mathcal{F}_{v\to u}(\widehat{m}^t)\;-\;\mathcal{F}_{v\to u}(m^*)\|_{L^2} \;\leq \sum_{w\in\mathcal{N}(v)\setminus\{u\}} L_{v\to u;w}\,\|\widehat{m}^t_{w\to v}-m^*_{w\to v}\|_{L^2},$$

for all $t=1,2,\ldots$.

Substituting the result of Lemma 8 in Equation (32) and performing some algebra, we find that

$$\begin{aligned}\sum_{\tau=0}^{t}\sum_{j=1}^{r}\left[b^{\tau}_{v\to u;j}-a^{*}_{v\to u;j}\right]^2 &\leq \sum_{\tau=0}^{t}\Big(\sum_{w\in\mathcal{N}(v)\setminus\{u\}} L_{v\to u;w}\,\|\widehat{m}^{\tau}_{w\to v}-m^*_{w\to v}\|_{L^2}\Big)^2\\ &\leq (d_v-1)\sum_{\tau=0}^{t}\sum_{w\in\mathcal{N}(v)\setminus\{u\}} L^2_{v\to u;w}\,\|\widehat{m}^{\tau}_{w\to v}-m^*_{w\to v}\|^2_{L^2},\end{aligned} \tag{33}$$

where d_v is the degree of node $v\in\mathcal{V}$. By definition, the message $\widehat{m}^{\tau}_{w\to v}$ is the L^2-projection of $m^{\tau}_{w\to v}$ onto $\mathcal{M}$. Since $m^*_{w\to v}\in\mathcal{M}$ and projection is non-expansive, we have

$$\begin{aligned}\|\widehat{m}^{\tau}_{w\to v}-m^*_{w\to v}\|^2_{L^2} &\leq \|m^{\tau}_{w\to v}-m^*_{w\to v}\|^2_{L^2}\\ &= \|\Delta^{\tau}_{w\to v}\|^2_{L^2}+\|A^r_{w\to v}\|^2_{L^2},\end{aligned} \tag{34}$$

where in the second step we have used the Pythagorean identity and recalled the definitions of estimation error as well as approximation error from (14) and (15). Substituting the inequality (34) into the bound (33) yields

$$\sum_{\tau=0}^{t}\sum_{j=1}^{r}\left[b^{\tau}_{v\to u;j}-a^{*}_{v\to u;j}\right]^2 \;\leq\; (d_v-1)\sum_{\tau=0}^{t}\sum_{w\in\mathcal{N}(v)\setminus\{u\}} L^2_{v\to u;w}\left(\|\Delta^{\tau}_{w\to v}\|^2_{L^2}+\|A^r_{w\to v}\|^2_{L^2}\right).$$

Therefore, introducing the convenient shorthand $\widetilde{L}_{v\to u,w}:=2\,(d_v-1)\,L^2_{v\to u;w}$, we have shown that

$$D^{t+1}_{v\to u} \;\leq\; \frac{1}{t+1}\sum_{\tau=0}^{t}\sum_{w\in\mathcal{N}(v)\setminus\{u\}}\widetilde{L}_{v\to u,w}\left(\|\Delta^{t}_{w\to v}\|^2_{L^2}+\|A^r_{w\to v}\|^2_{L^2}\right). \tag{35}$$

We make further use of this inequality shortly.

6.1.2 CONTROLLING THE STOCHASTIC TERM

We now turn to the stochastic part of the inequality (31). Our analysis is based on the following fact, proved in Appendix C:

Lemma 9 *For each $t \geq 0$, let $\mathcal{G}^t := \sigma(m^0, \ldots, m^t)$ be the σ-field generated by all messages through time t. Then for every fixed $j = 1, 2, \ldots, r$, the sequence $\zeta^{t+1}_{v\to u;j} = \widetilde{b}^{t+1}_{v\to u;j} - b^t_{v\to u;j}$ is a bounded martingale difference with respect to $\{\mathcal{G}^t\}_{t=0}^{\infty}$. In particular, we have $|\zeta^{t+1}_{v\to u;j}| \leq 2B_j$, where B_j was previously defined* (19).

Based on Lemma 9, standard martingale convergence results (Durrett, 1995) guarantee that for each $j = 1, 2, \ldots, r$, we have $\sum_{\tau=0}^{t} \zeta^{\tau+1}_{v\to u;j}/(t+1)$ converges to 0 almost surely (a.s.) as $t \to \infty$, and hence

$$S^{t+1}_{v\to u} \;=\; \frac{2}{(t+1)^2} \sum_{j=1}^{r} \Big\{ \sum_{\tau=0}^{t} \zeta^{\tau+1}_{v\to u;j} \Big\}^2 \;=\; 2 \sum_{j=1}^{r} \Big\{ \frac{1}{t+1} \sum_{\tau=0}^{t} \zeta^{\tau+1}_{v\to u;j} \Big\}^2 \;\xrightarrow{\text{a.s.}}\; 0. \tag{36}$$

Furthermore, we can apply the Azuma-Hoeffding inequality (Chung and Lu, 2006) in order to characterize the rate of convergence. For each $j = 1, 2, \ldots, r$, define the non-negative random variable $Z_j := \big\{ \sum_{\tau=0}^{t} \zeta^{\tau+1}_{v\to u;j} \big\}^2/(t+1)^2$. Since $|\zeta^{\tau+1}_{v\to u;j}| \leq 2B_j$, for any $\delta \geq 0$, we have

$$\mathbb{P}\big(Z_j \geq \delta\big) \;=\; \mathbb{P}\big(\sqrt{Z_j} \;\geq\; \sqrt{\delta}\big) \;\leq\; 2 \exp\bigg(- \frac{(t+1)\,\delta}{8\,B_j^2} \bigg),$$

for all $\delta > 0$. Moreover, Z_j is non-negative; therefore, integrating its tail bound we can compute the expectation

$$\mathbb{E}[Z_j] \;=\; \int_0^{\infty} \mathbb{P}\big(Z_j \geq \delta\big)\, d\delta \;\leq\; 2 \int_0^{\infty} \exp\bigg(- \frac{(t+1)\,\delta}{8\,B_j^2} \bigg)\, d\delta \;=\; \frac{16 B_j^2}{t+1},$$

and consequently

$$\mathbb{E}\big[|S^{t+1}_{v\to u}|\big] \;\leq\; \frac{32 \sum_{j=1}^{r} B_j^2}{t+1}.$$

6.1.3 ESTABLISHING CONVERGENCE

We now make use of the results established so far to prove the claims. Substituting the upper bound (35) on $D^{t+1}_{v\to u}$ into the decomposition (31) from Lemma 7, we find that

$$\|\Delta^{t+1}_{v\to u}\|_{L^2}^2 \;\leq\; \frac{1}{t+1} \sum_{\tau=0}^{t} \sum_{w \in \mathcal{N}(v)\setminus\{u\}} \widetilde{L}_{v\to u,w} \Big\{ \|\Delta^{\tau}_{w\to v}\|_{L^2}^2 + \|A^{r}_{w\to v}\|_{L^2}^2 \Big\} + S^{t+1}_{v\to u}. \tag{37}$$

For convenience, let us introduce the vector $T^{t+1} = \{T^{t+1}_{v\to u}, (v \to u) \in \vec{\mathcal{E}}\} \in \mathbb{R}^{|\vec{\mathcal{E}}|}$ with entries

$$T^{t+1}_{v\to u} := \frac{1}{t+1} \Big\{ \sum_{w \in \mathcal{N}(v)\setminus\{u\}} \widetilde{L}_{v\to u,w}\, \|\Delta^{0}_{w\to v}\|_{L^2}^2 \Big\} + S^{t+1}_{v\to u}. \tag{38}$$

Now define a matrix $N \in \mathbb{R}^{|\vec{\mathcal{E}}| \times |\vec{\mathcal{E}}|}$ with entries indexed by the directed edges and set to

$$N_{v \to u,\, w \to s} := \begin{cases} \widetilde{L}_{v \to u, w} & \text{if } s = v \text{ and } w \in \mathcal{N}(v) \setminus \{u\} \\ 0 & \text{otherwise.} \end{cases}$$

In terms of this matrix and the error terms $\rho^2(\,\cdot\,)$ previously defined in Equations (16) and (17), the scalar inequalities (37) can be written in the matrix form

$$\rho^2\big(\Delta^{t+1}\big) \preceq N\Big[\frac{1}{t+1}\sum_{\tau=1}^{t}\rho^2\big(\Delta^{\tau}\big)\Big] + N\rho^2\big(A^r\big) + T^{t+1}, \tag{39}$$

where $\preceq$ denotes the element-wise inequality based on the orthant cone.

From Lemma 1 in the paper by Noorshams and Wainwright (2013), we know that the matrix N is guaranteed to be nilpotent with degree ℓ equal to the graph diameter. Consequently, unwrapping the recursion (39) for a total of $\ell = \mathrm{diam}(\mathcal{G})$ times yields

$$\rho^2\big(\Delta^{t+1}\big) \preceq T_0^{t+1} + N\,T_1^{t+1} + \ldots + N^{\ell-1}\,T_{\ell-1}^{t+1} + (N + N^2 + \ldots + N^{\ell})\,\rho^2\big(A^r\big),$$

where we define $T_0^{t+1} \equiv T^{t+1}$, and then recursively $T_s^{t+1} := (\sum_{\tau=1}^{t} T_{s-1}^{\tau})/(t+1)$ for $s = 1, 2, \ldots, \ell-1$. By the nilpotency of N, we have the identity $I + N + \ldots + N^{\ell-1} = (I-N)^{-1}$; so we can further simplify the last inequality

$$\rho^2\big(\Delta^{t+1}\big) \preceq \sum_{s=0}^{\ell-1} N^s\,T_s^{t+1} + N\,(I-N)^{-1}\,\rho^2\big(A^r\big). \tag{40}$$

Recalling the definition $\mathcal{B} := \big\{e \in \mathbb{R}^{|\vec{\mathcal{E}}|} \mid |e| \preceq N(I-N)^{-1}\rho^2\big(A^r\big)\big\}$, inequality (40) implies that

$$\big|\rho^2\big(\Delta^{t+1}\big) - \Pi_{\mathcal{B}}\big(\rho^2\big(\Delta^{t+1}\big)\big)\big| \preceq \sum_{s=0}^{\ell-1} N^s\,T_s^{t+1}. \tag{41}$$

We now use the bound (41) to prove both parts of Theorem 2.

To prove the almost sure convergence claim in part (a), it suffices to show that for each $s = 0, 1, \ldots, \ell-1$, we have $T_s^t \xrightarrow{\text{a.s.}} 0$ as $t \to +\infty$. From Equation (36) we know $S_{v\to u}^{t+1} \to 0$ almost surely as $t \to \infty$. In addition, the first term in (38) is at most $O(1/t)$, so that also converges to zero as $t \to \infty$. Therefore, we conclude that $T_0^t \xrightarrow{\text{a.s.}} 0$ as $t \to \infty$. In order to extend this argument to higher-order terms, let us recall the following elementary fact from real analysis (Royden, 1988): for any sequence of real numbers $\{x^t\}_{t=0}^{\infty}$ such that $x^t \to 0$, then we also have $(\sum_{\tau=0}^{t} x^{\tau})/t \to 0$. In order to apply this fact, we observe that $T_0^t \xrightarrow{\text{a.s.}} 0$ means that for almost every sample point ω the deterministic sequence $\{T_0^{t+1}(\omega)\}_{t=0}^{\infty}$ converges to zero. Consequently, the above fact implies that $T_1^{t+1}(\omega) = (\sum_{\tau=1}^{t} T_0^{\tau}(\omega))/(t+1) \to 0$ as $t \to \infty$ for almost all sample points ω, which is equivalent to asserting that $T_1^t \xrightarrow{\text{a.s.}} \vec{0}$. Iterating the same argument, we establish $T_s^{t+1} \xrightarrow{\text{a.s.}} \vec{0}$ for all $s = 0, 1, \ldots, \ell-1$, thereby concluding the proof of Theorem 2(a).

Taking the expectation on both sides of the inequality (41) yields

$$\mathbb{E}\big[|\rho^2\big(\Delta^{t+1}\big) - \Pi_{\mathcal{B}}\big(\rho^2\big(\Delta^{t+1}\big)\big)|\big] \preceq \sum_{s=0}^{\ell-1} N^s\,\mathbb{E}[T_s^{t+1}] \tag{42}$$

so that it suffices to upper bound the expectations $\mathbb{E}[T_s^{t+1}]$ for $s = 0, 1, \ldots, \ell-1$. In Appendix D, we prove the following result:

Lemma 10 *Define the $|\vec{\mathcal{E}}|$-vector $\vec{v} := \{\sum_{j=1}^r B_j^2\}(4N\vec{1} + 32)$. Then for all $s = 0, 1, \ldots, \ell-1$ and $t = 0, 1, 2, \ldots$,*

$$\mathbb{E}[T_s^{t+1}] \preceq \frac{\vec{v}}{t+1}\left(\sum_{u=0}^{s}\frac{(\log(t+1))^u}{u!}\right). \tag{43}$$

Using this lemma, the proof of part (b) follows easily. In particular, substituting the bounds (43) into Equation (42) and doing some algebra yields

$$\begin{aligned}\mathbb{E}\big[|\rho^2(\Delta^{t+1}) - \Pi_{\mathcal{B}}(\rho^2(\Delta^{t+1}))|\big] &\preceq \sum_{s=0}^{\ell-1} N^s \sum_{u=0}^{s}\frac{(\log(t+1))^u}{u!}\Big(\frac{\vec{v}}{t+1}\Big)\\ &\preceq 3\sum_{s=0}^{\ell-1}(\log(t+1))^s N^s\Big(\frac{\vec{v}}{t+1}\Big)\\ &\preceq 3\,(I - \log(t+1)\,N)^{-1}\Big(\frac{\vec{v}}{t+1}\Big),\end{aligned}$$

where again we used the fact that $N^\ell = 0$.

6.2 General Graphs: Proof of Theorem 3

Recall the definition of the estimation error $\Delta^t_{v\to u}$ from (14). By Parseval's identity we know that $\|\Delta^t_{v\to u}\|_{L^2}^2 = \sum_{j=1}^r (a^t_{v\to u;j} - a^*_{v\to u;j})^2$. For convenience, we introduce the following shorthands for mean squared error on the directed edge $(v \to u)$

$$\bar{\rho}^2(\Delta^t_{v\to u}) := \mathbb{E}[\|\Delta^t_{v\to u}\|_{L^2}^2] = \mathbb{E}\Big[\sum_{j=1}^r (a^t_{v\to u;j} - a^*_{v\to u;j})^2\Big],$$

as well as the ℓ_∞ error

$$\bar{\rho}^2_{\max}(\Delta^t) := \max_{(v\to u)\in\vec{\mathcal{E}}} \mathbb{E}[\|\Delta^t_{v\to u}\|_{L^2}^2],$$

similarly defined for approximation error

$$\rho^2_{\max}(A^r) := \max_{(v\to u)\in\vec{\mathcal{E}}} \|A^r_{v\to u}\|_{L^2}^2.$$

Using the definition of $\bar{\rho}^2(\Delta^t_{v\to u})$, some algebra yields

$$\begin{aligned}\bar{\rho}^2(\Delta^{t+1}_{v\to u}) - \bar{\rho}^2(\Delta^t_{v\to u}) &= \mathbb{E}\Big[\sum_{j=1}^r \big(a^{t+1}_{v\to u;j} - a^*_{v\to u;j}\big)^2 - \sum_{j=1}^r \big(a^t_{v\to u;j} - a^*_{v\to u;j}\big)^2\Big]\\ &= \mathbb{E}\Big[\sum_{j=1}^r \big\{a^{t+1}_{v\to u;j} - a^t_{v\to u;j}\big\}\big\{\big(a^{t+1}_{v\to u;j} - a^t_{v\to u;j}\big) + 2\big(a^t_{v\to u;j} - a^*_{v\to u;j}\big)\big\}\Big].\end{aligned}$$

From the update Equation (13), we have

$$a^{t+1}_{v\to u;j} - a^t_{v\to u;j} = \eta^t\,\big(\widetilde{b}^{t+1}_{v\to u;j} - a^t_{v\to u;j}\big),$$

and hence

$$\bar{\rho}^2(\Delta^{t+1}_{v\to u}) - \bar{\rho}^2(\Delta^t_{v\to u}) \;=\; U^t_{v\to u} + V^t_{v\to u}, \tag{44}$$

where

$$U^t_{v\to u} \;:=\; (\eta^t)^2 \sum_{j=1}^{r} \mathbb{E}\Big[\big(\widetilde{b}^{t+1}_{v\to u;j} - a^t_{v\to u;j}\big)^2\Big], \quad \text{and}$$
$$V^t_{v\to u} \;:=\; 2\eta^t \sum_{j=1}^{r} \mathbb{E}\Big[\big(\widetilde{b}^{t+1}_{v\to u;j} - a^t_{v\to u;j}\big)\big(a^t_{v\to u;j} - a^*_{v\to u;j}\big)\Big].$$

The following lemma, proved in Appendix E, provides upper bounds on these two terms.

Lemma 11 *For all iterations $t = 0, 1, 2, \ldots$, we have*

$$U^t_{v\to u} \;\le\; 4(\eta^t)^2 \sum_{j=1}^{r} B_j^2, \quad \textit{and} \tag{45}$$
$$V^t_{v\to u} \;\le\; \eta^t\big(1-\frac{\gamma}{2}\big)\rho^2_{max}(A^r) + \eta^t\big(1-\frac{\gamma}{2}\big)\bar{\rho}^2_{max}(\Delta^t) - \eta^t(1+\frac{\gamma}{2})\bar{\rho}^2(\Delta^t_{v\to u}). \tag{46}$$

We continue upper-bounding $\bar{\rho}^2(\Delta^{t+1}_{v\to u})$ by substituting the results of Lemma 11 into Equation (44), thereby obtaining

$$\begin{aligned}
\bar{\rho}^2(\Delta^{t+1}_{v\to u}) \;&\le\; 4(\eta^t)^2 \sum_{j=1}^{r} B_j^2 + \eta^t\big(1-\frac{\gamma}{2}\big)\,\rho^2_{\max}(A^r) + \eta^t\big(1-\frac{\gamma}{2}\big)\,\bar{\rho}^2_{\max}(\Delta^t) \\
&\qquad + \Big\{1 - \eta^t\big(1+\frac{\gamma}{2}\big)\Big\}\bar{\rho}^2(\Delta^t_{v\to u}) \\
&\le\; 4(\eta^t)^2 \sum_{j=1}^{r} B_j^2 + \eta^t\big(1-\frac{\gamma}{2}\big)\rho^2_{\max}(A^r) + \big(1-\eta^t\gamma\big)\bar{\rho}^2_{\max}(\Delta^t).
\end{aligned}$$

Since this equation holds for all directed edges $(v \to u)$, taking the maximum over the left-hand side yields the recursion

$$\bar{\rho}^2_{\max}(\Delta^{t+1}) \;\le\; (\eta^t)^2\,\bar{B}^2 + \eta^t\big(1-\frac{\gamma}{2}\big)\,\rho^2_{\max}(A^r) + \big(1-\eta^t\gamma\big)\,\bar{\rho}^2_{\max}(\Delta^t),$$

where we have introduced the shorthand $\bar{B}^2 = 4\sum_{j=1}^{r} B_j^2$. Setting $\eta^t = 1/(\gamma(t+1))$ and unwrapping this recursion, we find that

$$\begin{aligned}
\bar{\rho}^2_{\max}(\Delta^{t+1}) \;&\le\; \frac{\bar{B}^2}{\gamma^2}\sum_{\tau=1}^{t+1}\frac{1}{\tau(t+1)} + \frac{2-\gamma}{2\gamma}\rho^2_{\max}(A^r) \\
&\le\; \frac{2\bar{B}^2}{\gamma^2}\,\frac{\log(t+1)}{t+1} + \frac{1}{\gamma}\rho^2_{\max}(A^r),
\end{aligned}$$

which establishes the claim.

6.3 Complexity versus Accuracy: Proof of Theorem 4

As discussed earlier, each iteration of the SOSMP algorithm requires $O(r)$ operations per edge. Consequently, it suffices to show that running the algorithm with $r = r^*$ coefficients for $(\sum_{j=1}^{r} \lambda_j^2)(1/\delta)\log(1/\delta)$ iterations suffices to achieve mean-squared error of the order of δ.

The bound (21) consists of two terms. In order to characterize the first term (estimation error), we need to bound B_j defined in (19). Using the orthonormality of the basis functions and the fact that the supremum is attainable over the compact space $\mathcal{X}$, we obtain

$$B_j \;=\; \max_{(v\to u)\in\vec{\mathcal{E}}} \sup_{y\in\mathcal{X}} \frac{\lambda_j \phi_j(y)}{\int_{\mathcal{X}} \psi_{uv}(x,y)\mu(dx)} \;=\; O(\lambda_j).$$

Therefore, the estimation error decays at the rate $O\big((\sum_{j=1}^{r} \lambda_j^2)\,(\log t/t)\big)$, so that a total of $t = O\big((\sum_{j=1}^{r} \lambda_j^2)(1/\delta)\log(1/\delta)\big)$ iterations are sufficient to reduce it to $O(\delta)$.

The second term (approximation error) in the bound (21) depends only on the choice of r, and in particular on the r-term approximation error $\|A^r_{v\to u}\|^2_{L^2} = \|m^*_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2}$. To bound this term, we begin by representing $m^*_{v\to u}$ in terms of the basis expansion $\sum_{j=1}^{\infty} a^*_j \phi_j$. By the Pythagorean theorem, we have

$$\|m^*_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2} = \sum_{j=r+1}^{\infty} (a^*_j)^2. \tag{47}$$

Our first claim is that $\sum_{j=1}^{\infty} (a^*_j)^2/\lambda_j < \infty$. Indeed, since m^* is a fixed point of the message update equation, we have

$$m^*_{v\to u}(\cdot) \;\propto\; \int_{\mathcal{X}} \psi_{uv}(\cdot, y)\, M(y)\, \mu(dy),$$

where $M(\cdot) := \psi_v(\cdot) \prod_{w\in\mathcal{N}(v)\setminus\{u\}} m^*_{w\to v}(\cdot)$. Exchanging the order of integrations using Fubini's theorem, we obtain

$$a^*_j \;=\; \langle m^*_{v\to u}, \phi_j\rangle_{L^2} \;\propto\; \int_{\mathcal{X}} \langle \phi_j(\cdot), \psi_{uv}(\cdot, y)\rangle_{L^2}\, M(y)\, \mu(dy). \tag{48}$$

By the eigenexpansion of ψ_{uv}, we have

$$\langle \phi_j(\cdot), \psi_{uv}(\cdot, y)\rangle_{L^2} \;=\; \sum_{k=1}^{\infty} \lambda_k \langle \phi_j, \phi_k\rangle_{L^2}\, \phi_k(y) \;=\; \lambda_j\, \phi_j(y).$$

Substituting back into our initial Equation (48), we find that

$$a^*_j \;\propto\; \lambda_j \int_{\mathcal{X}} \phi_j(y)\, M(y)\, \mu(dy) \;=\; \lambda_j\, \widetilde{a}_j,$$

where $\widetilde{a}_j$ are the basis expansion coefficients of M. Since the space $\mathcal{X}$ is compact, one can see that $M \in L^2(\mathcal{X})$, and hence $\sum_{j=1}^{\infty} \widetilde{a}_j^2 < \infty$. Therefore, we have

$$\sum_{j=1}^{\infty} \frac{(a^*_j)^2}{\lambda_j} \;\propto\; \sum_{j=1}^{\infty} \lambda_j\, \widetilde{a}_j^2 \;<\; +\infty,$$

where we used the fact that $\sum_{j=1}^{\infty} \lambda_j < \infty$.

We now use this bound to control the approximation error (47). For any $r = 1, 2, \ldots$, we have

$$\sum_{j=r+1}^{\infty} (a_j^*)^2 \;=\; \sum_{j=r+1}^{\infty} \lambda_j \frac{(a_j^*)^2}{\lambda_j} \;\leq\; \lambda_r \sum_{j=r+1}^{\infty} \frac{(a_j^*)^2}{\lambda_j} \;=\; O(\lambda_r),$$

using the non-increasing nature of the sequence $\{\lambda_j\}_{j=1}^{\infty}$. Consequently, by definition of r^* (22), we have

$$\|m^*_{v\to u} - \Pi^{r^*}(m^*_{v\to u})\|^2_{L^2} = O(\delta),$$

as claimed.

7. Conclusion

Belief propagation is a widely used message-passing algorithm for computing (approximate) marginals in graphical models. In this paper, we have presented and analyzed the SOSMP algorithm for running BP in models with continuous variables. It is based on two forms of approximation: a *deterministic approximation* that involves projecting messages onto the span of r basis functions, and a *stochastic approximation* that involves approximating basis coefficients via Monte Carlo techniques and damped updates. These approximations, while leading to an algorithm with substantially reduced complexity, are also controlled: we provide upper bounds on the convergence of the stochastic error, showing that it goes to zero as $O(\log t/t)$ with the number of iterations, and also control on the deterministic error. For graphs with relatively smooth potential functions, as reflected in the decay rate of their basis coefficients, we provided a quantitative bound on the total number of basic arithmetic operations required to compute the BP fixed point to within δ-accuracy. We illustrated our theoretical predictions using experiments on simulated graphical models, as well as in a real-world instance of optical flow estimation.

Our work leaves open a number of interesting questions. First, although we have focused exclusively on models with pairwise interactions, it should be possible to develop forms of SOSMP for higher-order factor graphs. Second, the bulk of our analysis was performed under a type of contractivity condition, as has been used in past works (Tatikonda and Jordan, 2002; Ihler et al., 2005; Mooij and Kappen, 2007; Roosta et al., 2008) on convergence of the standard BP updates. However, we suspect that this condition might be somewhat relaxed, and doing so would demonstrate applicability of the SOSMP algorithm to a larger class of graphical models. Finally, it would be interesting to see if the ideas presented in this work can be applied to other graph-based learning problems.

Acknowledgments

This work was partially supported by Office of Naval Research MURI grant N00014-11-1-0688 to MJW. The authors thank Erik Sudderth for helpful discussions and pointers to the literature.

Appendix A. Proof of Lemma 7

Subtracting $a^*_{v\to u;j}$ from both sides of the update (13) in Step 2(c), we obtain

$$a^{t+1}_{v\to u;j} - a^*_{v\to u;j} \;=\; (1-\eta^t)\left[a^t_{v\to u;j} - a^*_{v\to u;j}\right] + \eta^t\left[b^t_{v\to u;j} - a^*_{v\to u;j}\right] + \eta^t\,\zeta^{t+1}_{v\to u;j}. \tag{49}$$

Setting $\eta^t = 1/(t+1)$ and unwrapping the recursion (49) then yields

$$a^{t+1}_{v\to u;j} - a^*_{v\to u;j} \;=\; \frac{1}{t+1}\sum_{\tau=0}^{t}\left[b^\tau_{v\to u;j} - a^*_{v\to u;j}\right] + \frac{1}{t+1}\sum_{\tau=0}^{t}\zeta^{\tau+1}_{v\to u;j}.$$

Squaring both sides of this equality and using the upper bound $(a+b)^2 \le 2a^2+2b^2$, we obtain

$$\left(a^{t+1}_{v\to u;j} - a^*_{v\to u;j}\right)^2 \;\le\; \frac{2}{(t+1)^2}\left\{\sum_{\tau=0}^{t}\left[b^\tau_{v\to u;j} - a^*_{v\to u;j}\right]\right\}^2 + \frac{2}{(t+1)^2}\left\{\sum_{\tau=0}^{t}\zeta^{\tau+1}_{v\to u;j}\right\}^2.$$

Summing over indices $j = 1,2,\ldots,r$ and recalling the expansion (30), we find that

$$\begin{aligned}\|\Delta^t_{v\to u}\|^2_{L^2} \;&\le\; \sum_{j=1}^{r}\left\{\frac{2}{(t+1)^2}\left\{\sum_{\tau=0}^{t}\left[b^\tau_{v\to u;j} - a^*_{v\to u;j}\right]\right\}^2 + \frac{2}{(t+1)^2}\left\{\sum_{\tau=0}^{t}\zeta^{\tau+1}_{v\to u;j}\right\}^2\right\} \\ &\overset{\text{(i)}}{\le} \underbrace{\frac{2}{(t+1)}\sum_{j=1}^{r}\sum_{\tau=0}^{t}\left[b^\tau_{v\to u;j} - a^*_{v\to u;j}\right]^2}_{\text{Deterministic term } D^{t+1}_{v\to u}} + \underbrace{\frac{2}{(t+1)^2}\sum_{j=1}^{r}\left\{\sum_{\tau=0}^{t}\zeta^{\tau+1}_{v\to u;j}\right\}^2}_{\text{Stochastic term } S^{t+1}_{v\to u}}.\end{aligned}$$

Here step (i) follows from the elementary inequality

$$\left\{\sum_{\tau=0}^{t}\left[b^\tau_{v\to u;j} - a^*_{v\to u;j}\right]\right\}^2 \;\le\; (t+1)\sum_{\tau=0}^{t}\left[b^\tau_{v\to u;j} - a^*_{v\to u;j}\right]^2.$$

Appendix B. Proof of Lemma 8

Recall the probability density

$$[p_{v\to u}(m)](\cdot) \;\propto\; \beta_{v\to u}(\cdot)\prod_{w\in\mathcal{N}(v)\setminus\{u\}} m_{w\to v}(\cdot),$$

defined in Step 2 of the SOSMP algorithm. Using this shorthand notation, the claim of Lemma 1 can be re-written as $[\mathcal{F}_{v\to u}(m)](x) = \langle \Gamma_{uv}(x,\cdot), [p_{v\to u}(m)](\cdot)\rangle_{L^2}$. Therefore, applying the Cauchy-Schwartz inequality yields

$$|[\mathcal{F}_{v\to u}(m)](x) - [\mathcal{F}_{v\to u}(m')](x)|^2 \;\le\; \|\Gamma_{uv}(x,\cdot)\|^2_{L^2}\;\|p_{v\to u}(m) - p_{v\to u}(m')\|^2_{L^2}.$$

Integrating both sides of the previous inequality over $\mathcal{X}$ and taking square roots yields

$$\|\mathcal{F}_{v\to u}(m) - \mathcal{F}_{v\to u}(m')\|_{L^2} \;\le\; C_{uv}\,\|p_{v\to u}(m) - p_{v\to u}(m')\|_{L^2},$$

where we have denoted the constant $C_{uv} := \left(\int_{\mathcal{X}} |\Gamma_{uv}(x,y)|^2\mu(dy)\mu(dx)\right)^{1/2}$.

The next step is to upper bound the term $\|p_{v\to u}(m) - p_{v\to u}(m')\|_{L^2}$. In order to do so, we first show that $p_{v\to u}(m)$ is a Frechet differentiable[4] operator on the space $\mathcal{M}' := \text{convhull}\{m^*, \oplus_{(v\to u)\in\vec{\mathcal{E}}} \mathcal{M}'_{v\to u}\}$, where

$$\mathcal{M}'_{v\to u} := \left\{ \widehat{m}_{v\to u} \,\middle|\, \widehat{m}_{v\to u} = \Big[\mathbb{E}_{Y\sim f}\big[\Pi^r\big(\Gamma_{uv}(\cdot, Y)\big)\big]\Big]_+, \text{ for some probability density } f \right\},$$

denotes the space of all feasible SOSMP messages on the directed edge $(v \to u)$. Doing some calculus using the chain rule, we calculate the partial directional (Gateaux) derivative of the operator $p_{v\to u}(m)$ with respect to the function $m_{w\to v}$. More specifically, for an arbitrary function $h_{w\to v}$, we have

$$\begin{aligned}
[\mathcal{D}_w\, p_{v\to u}(m)](h_{w\to v}) \;=\; & \frac{\beta_{v\to u}\prod_{s\in\mathcal{N}(v)\setminus\{u,w\}} m_{s\to v}}{\langle M_{v\to u}, \beta_{v\to u}\rangle_{L^2}}\, h_{w\to v} \\
& - \frac{\beta_{v\to u} M_{v\to u}}{\langle M_{v\to u}, \beta_{v\to u}\rangle^2_{L^2}}\, \langle h_{w\to v}, \beta_{v\to u} \prod_{s\in\mathcal{N}(v)\setminus\{u,w\}} m_{s\to v}\rangle_{L^2},
\end{aligned}$$

where $M_{v\to u} = \prod_{w\in\mathcal{N}(v)\setminus\{u\}} m_{w\to v}$. Clearly the Gateaux derivative is linear and continuous. It is also bounded as will be shown now. Massaging the operator norm's definition, we obtain

$$\begin{aligned}
\sup_{m\in\mathcal{M}'} \|\!|\mathcal{D}_w\, p_{v\to u}(m)|\!\|_2 \;=\; & \sup_{m\in\mathcal{M}'} \sup_{h_{w\to v}\in\mathcal{M}'_{w\to v}} \frac{\|[\mathcal{D}_w\, p_{v\to u}(m)](h_{w\to v})\|_{L^2}}{\|h_{w\to v}\|_{L^2}} \\
\leq\; & \sup_{m\in\mathcal{M}'} \frac{\sup_{x\in\mathcal{X}}\, \beta_{v\to u}(x)\prod_{s\in\mathcal{N}(v)\setminus\{u,w\}} m_{s\to v}(x)}{\langle M_{v\to u}, \beta_{v\to u}\rangle_{L^2}} \\
& + \sup_{m\in\mathcal{M}'} \frac{\|\beta_{v\to u} M_{v\to u}\|_{L^2}\, \|\beta_{v\to u}\prod_{s\in\mathcal{N}(v)\setminus\{u,w\}} m_{s\to v}\|_{L^2}}{\langle M_{v\to u}, \beta_{v\to u}\rangle^2_{L^2}}. \qquad (50)
\end{aligned}$$

Since the space $\mathcal{X}$ is compact, the continuous functions $\beta_{v\to u}$ and $m_{s\to v}$ achieve their maximum over $\mathcal{X}$. Therefore, the numerator of (50) is bounded and we only need to show that the denominator is bounded away from zero.

For an arbitrary message $m_{v\to u} \in \mathcal{M}'_{v\to u}$ there exist $0 < \alpha < 1$ and a bounded probability density f so that

$$m_{v\to u}(x) \;=\; \alpha\, m^*_{v\to u}(x) \;+\; (1-\alpha)\Big[\mathbb{E}_{Y\sim f}\big[\widetilde{\Gamma}_{uv}(x, Y)\big]\Big]_+,$$

4. For the convenience of the reader, we state the notions of Gateaux and Frechet differentiability (Clarke, 2013; Fabian et al., 2011). For normed spaces X and Y, let U be an open subset of X, and let $F : X \to Y$ be an operator. For $x \in U$ and $z \in X$, the F is Gateaux differentiable at x in the direction z if and only if the following limit exists

$$[\mathcal{D}F(x)](z) \;:=\; \lim_{t\to 0} \frac{F(x+tz) - F(x)}{t} \;=\; \frac{d}{dt} F(x+tz)\,|_{t=0}\,.$$

Moreover, the operator F is Frechet differentiable at x if there exists a bounded linear operator $\mathcal{D}F(x) : X \to Y$ such that

$$\lim_{z\to 0} \frac{\|F(x+z) - F(x) - [\mathcal{D}F(x)](z)\|}{\|z\|} \;=\; 0.$$

where we have introduced the shorthand $\widetilde{\Gamma}_{uv}(\cdot,y) := \Pi^r(\Gamma_{uv}(\cdot,y))$. According to Lemma 1, we know $m^*_{v\to u} = \mathbb{E}_Y[\Gamma_{uv}(\cdot,Y)]$, where $Y \sim p_{v\to u}(m^*)$. Therefore, denoting $p^* = p_{v\to u}(m^*)$, we have

$$\begin{aligned} m_{v\to u}(x) \;&\geq\; \alpha\,\mathbb{E}_{Y\sim p^*}[\Gamma_{uv}(x,Y)] + (1-\alpha)\,\mathbb{E}_{Y\sim f}[\widetilde{\Gamma}_{uv}(x,Y)] \\ &=\; \mathbb{E}_{Y\sim(\alpha p^*+(1-\alpha)f)}[\widetilde{\Gamma}_{uv}(x,Y)] + \alpha\,\mathbb{E}_{Y\sim p^*}[\Gamma_{uv}(x,Y) - \widetilde{\Gamma}_{uv}(x,Y)]. \end{aligned} \tag{51}$$

On the other hand, since $\mathcal{X}$ is compact, we can exchange the order of expectation and projection using Fubini's theorem to obtain

$$\mathbb{E}_{Y\sim p^*}[\Gamma_{uv}(\cdot,Y) - \widetilde{\Gamma}_{uv}(\cdot,Y)] \;=\; m^*_{v\to u} - \Pi^r(m^*_{v\to u}) \;=\; A^r_{v\to u}.$$

Substituting the last equality into the bound (51) yields

$$m_{v\to u}(x) \;\geq\; \inf_{y\in\mathcal{X}} \widetilde{\Gamma}_{uv}(x,y) - |A^r_{v\to u}(x)|.$$

Recalling the assumption (18), one can conclude that the right hand side of the above inequality is positive for all directed edges $(v\to u)$. Therefore, the denominator of the expression (50) is bounded away from zero and more importantly $\sup_{m\in\mathcal{M}'} |||\mathcal{D}_w\, p_{v\to u}(m)|||_2$ is attainable.

Since the derivative is a bounded, linear, and continuous operator, the Gateaux and Frechet derivatives coincides and we can use Proposition 2 (Luenberger, 1969, page 176) to obtain the following upper bound

$$\|p_{v\to u}(m) - p_{v\to u}(m')\|_{L^2} \;\leq \sum_{w\in\mathcal{N}(v)\setminus\{u\}} \sup_{0\leq\alpha\leq 1} |||\mathcal{D}_w\, p_{v\to u}(m' + \alpha\,(m-m'))|||_2\, \|m_{w\to v} - m'_{w\to v}\|_{L^2}.$$

Setting $L_{v\to u;w} := C_{uv}\, \sup_{m\in\mathcal{M}'} |||\mathcal{D}_w\, p_{v\to u}(m)|||_2$ and putting the pieces together yields

$$\|\mathcal{F}_{v\to u}(m) - \mathcal{F}_{v\to u}(m')\|_{L^2} \;\leq \sum_{w\in\mathcal{N}(v)\setminus\{u\}} L_{v\to u;w}\, \|m_{w\to v} - m'_{w\to v}\|_{L^2},$$

for all $m, m' \in \mathcal{M}'$.

The last step of the proof is to verify that $m^* \in \mathcal{M}'$, and $\widehat{m}^t \in \mathcal{M}'$ for all $t = 1,2,\ldots$. By definition we have $m^* \in \mathcal{M}'$. On the other hand, unwrapping the update (13) we obtain

$$\begin{aligned} a^t_{v\to u;j} \;&= \frac{1}{t}\sum_{\tau=0}^{t-1} \widetilde{b}^{\tau+1}_{v\to u;j} \\ &= \frac{1}{t}\sum_{\tau=0}^{t-1}\frac{1}{k}\sum_{\ell=1}^{k}\int_{\mathcal{X}} \Gamma_{uv}(x, Y^{\tau+1}_\ell)\,\phi_j(x)\,\mu(dx) \\ &= \int_{\mathcal{X}} \mathbb{E}_{Y\sim\hat{p}}[\Gamma_{uv}(x,Y)]\,\phi_j(x)\,\mu(dx), \end{aligned}$$

where $\hat{p}$ denotes the empirical probability density. Therefore, $m^t_{v\to u} = \sum_{j=1}^r a^t_{v\to u;j}\,\phi_j$ is equal to $\Pi^r(\mathbb{E}_{Y\sim\hat{p}}[\Gamma_{uv}(\cdot,Y)])$, thereby completing the proof.

Appendix C. Proof of Lemma 9

We begin by taking the conditional expectation of $\widetilde{b}^{t+1}_{v\to u;j}$, previously defined (12), given the filteration $\mathcal{G}^t$ and with respect to the random samples $\{Y^{t+1}_1,\ldots,Y^{t+1}_k\} \overset{\text{i.i.d.}}{\sim} [p_{v\to u}(\widehat{m})](\cdot)$. Exchanging the order of expectation and integral[5] and exploiting the result of Lemma 1, we obtain

$$\mathbb{E}[\widetilde{b}^{t+1}_{v\to u;j} \,|\, \mathcal{G}^t] \;=\; \int_{\mathcal{X}} [\mathcal{F}_{v\to u}(\widehat{m}^t)](x)\,\phi_j(x)\,\mu(dx) \;=\; b^t_{v\to u;j}, \tag{52}$$

and hence $\mathbb{E}[\zeta^{t+1}_{v\to u;j} \,|\, \mathcal{G}^t] = 0$, for all $j = 1,2,\ldots,r$ and all directed edges $(v\to u)\in\vec{\mathcal{E}}$. Also it is clear that $\zeta^{t+1}_{v\to u;j}$ is $\mathcal{G}^t$-measurable. Therefore, $\{\zeta^{\tau+1}_{v\to u;j}\}_{\tau=0}^{\infty}$ forms a martingale difference sequence with respect to the filtration $\{\mathcal{G}^\tau\}_{\tau=0}^{\infty}$. On the other hand, recalling the bound (19), we have

$$|\widetilde{b}^{t+1}_{v\to u;j}| \;\le\; \frac{1}{k}\sum_{\ell=1}^{k} |\langle \Gamma_{uv}(\cdot,Y_\ell),\phi_j\rangle_{L^2}| \;\le\; B_j.$$

Moreover, exploiting the result of Lemma 1 and exchanging the order of the integration and expectation once more yields

$$|b^t_{v\to u;j}| \;=\; |\langle \mathbb{E}_Y[\Gamma_{uv}(\cdot,Y)],\phi_j\rangle_{L^2}| \;=\; |\mathbb{E}_Y[\langle \Gamma_{uv}(\cdot,Y),\phi_j\rangle_{L^2}]| \;\le\; B_j, \tag{53}$$

where we have $Y\sim[p_{v\to u}(\widehat{m}^t)](y)$. Therefore, the martingale difference sequence is bounded, in particular with

$$|\zeta^{t+1}_{v\to u;j}| \;\le |\widetilde{b}^{t+1}_{v\to u;j}| + |\,b^t_{v\to u;j}| \;\le\; 2B_j.$$

Appendix D. Proof of Lemma 10

We start by uniformly upper-bounding the terms $\mathbb{E}[|T^{t+1}_{v\to u}|]$. To do so we first need to bound $\|\Delta^t_{v\to u}\|_{L^2}$. By definition we know $\|\Delta^t_{v\to u}\|^2_{L^2} = \sum_{j=1}^r [a^t_{v\to u;j} - a^*_{v\to u;j}]^2$; therefore we only need to control the terms $a^t_{v\to u;j}$ and $a^*_{v\to u;j}$ for $j = 1,2,\ldots,r$.

By construction, we always have $|\widetilde{b}^{t+1}_{v\to u;j}| \le B_j$ for all iterations $t = 0,1,\ldots$. Also, assuming that $|a^0_{v\to u;j}| \le B_j$, without loss of generality, a simple induction using the update Equation (13) shows that $|a^t_{v\to u;j}| \le B_j$ for all t. Moreover, using a similar argument leading to (53), we obtain

$$|a^*_{v\to u;j}| \;=\; |\langle \mathbb{E}_Y[\Gamma_{uv}(\cdot,Y)],\phi_j\rangle_{L^2}| \;=\; |\mathbb{E}_Y[\langle \Gamma_{uv}(\cdot,Y),\phi_j\rangle_{L^2}]| \;\le\; B_j,$$

where we have $Y\sim[p_{v\to u}(m^*)](y)$. Therefore, putting the pieces together, recalling the definition (38) of $T^{t+1}_{v\to u}$ yields

$$\mathbb{E}[|T^{t+1}_{v\to u}|] \;\le\; \frac{4}{t+1}\sum_{w\in\mathcal{N}(v)\setminus\{u\}} \widetilde{L}_{v\to u,w}\sum_{j=1}^{r} B_j^2 + \frac{32}{t+1}\sum_{j=1}^{r} B_j^2.$$

5. Since $\Gamma_{uv}(x,y)\phi_i(x)[p_{v\to u}(\widehat{m}^t)](y)$ is absolutely integrable, we can exchange the order of the integrals using Fubini's theorem.

Concatenating the previous scalar inequalities yields $\mathbb{E}[T_0^{t+1}] \preceq \vec{v}/(t+1)$, for all $t \geq 0$, where we have defined the $|\vec{\mathcal{E}}|$-vector $\vec{v} := \{\sum_{j=1}^r B_j^2\}(4N\vec{1}+32)$. We now show, using an inductive argument, that

$$\mathbb{E}[T_s^{t+1}] \;\preceq\; \frac{\vec{v}}{t+1} \sum_{u=0}^{s} \frac{(\log(t+1))^u}{u!},$$

for all $s = 0,1,2,\ldots$ and $t = 0,1,2,\ldots$. We have already established the base case $s = 0$. For some $s > 0$, assume that the claim holds for $s-1$. By the definition of T_s^{t+1}, we have

$$\begin{aligned}\mathbb{E}[T_s^{t+1}] \;&= \frac{1}{t+1} \sum_{\tau=1}^{t} \mathbb{E}[T_{s-1}^{\tau}] \\ &\preceq \frac{\vec{v}}{t+1} \sum_{\tau=1}^{t} \Big\{ \frac{1}{\tau} + \sum_{u=1}^{s-1} \frac{(\log\tau)^u}{u!\,\tau} \Big\},\end{aligned}$$

where the inequality follows from the induction hypothesis. We now make note of the elementary inequalities $\sum_{\tau=1}^t 1/\tau \leq 1 + \log t$, and

$$\sum_{\tau=1}^{t} \frac{(\log\tau)^u}{u!\,\tau} \;\leq\; \int_1^t \frac{(\log x)^u}{u!\,x}\,dx \;=\; \frac{(\log t)^{(u+1)}}{(u+1)!}, \qquad \text{for all } u \geq 1$$

from which the claim follows.

Appendix E. Proof of Lemma 11

By construction, we always have $|\widetilde{b}_{v\to u;j}^{t+1}| \leq B_j$ for all iterations $t = 0,1,2,\ldots$. Moreover, assuming $|a_{v\to u;j}^0| \leq B_j$, without loss of generality, a simple induction on the update equation shows that $|a_{v\to u;j}^t| \leq B_j$ for all iterations $t = 0,1,\ldots$. On this basis, we find that

$$U_{v\to u}^t \;=\; (\eta^t)^2 \sum_{j=1}^{r} \mathbb{E}\big[\big(\widetilde{b}_{v\to u;j}^{t+1} - a_{v\to u;j}^t\big)^2\big] \;\leq\; 4(\eta^t)^2 \sum_{j=1}^{r} B_j^2,$$

which establishes the bound (45).

It remains to establish the bound (46) on $V_{v\to u}^t$. We first condition on the σ-field $\mathcal{G}^t = \sigma(m^0,\ldots,m^t)$ and take expectations over the remaining randomness, thereby obtaining

$$\begin{aligned}V_{v\to u}^t \;&=\; 2\eta^t\,\mathbb{E}\Big[\mathbb{E}\big[\sum_{j=1}^{r}\big(\widetilde{b}_{v\to u;j}^{t+1} - a_{v\to u;j}^t\big)\big(a_{v\to u;j}^t - a_{v\to u;j}^*\big)\;\big|\;\mathcal{G}^t\big]\Big] \\ &=\; 2\eta^t\,\mathbb{E}\Big[\sum_{j=1}^{r}\big(b_{v\to u;j}^t - a_{v\to u;j}^t\big)\big(a_{v\to u;j}^t - a_{v\to u;j}^*\big)\Big],\end{aligned}$$

where $\{b_{v\to u;j}^t\}_{j=1}^{\infty}$ are the expansion coefficients of the function $\mathcal{F}_{v\to u}(\widehat{m}^t)$ (i.e., $b_{v\to u;j}^t = \langle \mathcal{F}_{v\to u}(\widehat{m}^t), \phi_j\rangle_{L^2}$), and we have recalled the result $\mathbb{E}[\widetilde{b}_{v\to u;j}^{t+1}|\mathcal{G}^t] = b_{v\to u;j}^t$ from (52). By Parseval's identity, we have

$$\begin{aligned}T \;&:=\; \sum_{j=1}^{r}\big(b_{v\to u;j}^t - a_{v\to u;j}^t\big)\big(a_{v\to u;j}^t - a_{v\to u;j}^*\big) \\ &=\; \langle \Pi^r(\mathcal{F}_{v\to u}(\widehat{m}^t)) - m_{v\to u}^t,\, m_{v\to u}^t - \Pi^r(m_{v\to u}^*)\rangle_{L^2}.\end{aligned}$$

Here we have used the basis expansions

$$m^t_{v\to u} = \sum_{j=1}^{r} a^t_{v\to u;j}\phi_j, \quad \text{and} \quad \Pi^r(m^*_{v\to u}) = \sum_{j=1}^{r} a^*_{v\to u;j}\phi_j.$$

Since $\Pi^r(m^t_{v\to u}) = m^t_{v\to u}$ and $\mathcal{F}_{v\to u}(m^*) = m^*_{v\to u}$, we have

$$\begin{aligned} T &= \langle \Pi^r\big(\mathcal{F}_{v\to u}(\widehat{m}^t) - \mathcal{F}_{v\to u}(m^*)\big), m^t_{v\to u} - \Pi^r(m^*_{v\to u})\rangle_{L^2} - \|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2} \\ &\overset{(i)}{\leq} \|\Pi^r\big(\mathcal{F}_{v\to u}(\widehat{m}^t) - \mathcal{F}_{v\to u}(m^*)\big)\|_{L^2}\ \|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|_{L^2} - \|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2} \\ &\overset{(ii)}{\leq} \|\mathcal{F}_{v\to u}(\widehat{m}^t) - \mathcal{F}_{v\to u}(m^*)\|_{L^2}\,\|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|_{L^2} - \|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2}. \end{aligned}$$

where step (i) uses the Cauchy-Schwarz inequality, and step (ii) uses the non-expansivity of projection. Applying the contraction condition (20), we obtain

$$\begin{aligned} T &\leq \big(1-\frac{\gamma}{2}\big)\sqrt{\frac{\sum_{w\in\mathcal{N}(v)\setminus\{u\}} \|\widehat{m}^t_{w\to v} - m^*_{w\to v}\|^2_{L^2}}{|\mathcal{N}(v)|-1}}\ \|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|_{L^2} \\ &\quad - \|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2} \\ &\leq \big(1-\frac{\gamma}{2}\big)\Big\{\frac{1}{2}\,\frac{\sum_{w\in\mathcal{N}(v)\setminus\{u\}}\|m^t_{w\to v} - m^*_{w\to v}\|^2_{L^2}}{|\mathcal{N}(v)|-1} + \frac{1}{2}\,\|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2}\Big\} \\ &\quad - \|m^t_{v\to u} - \Pi^r(m^*_{v\to u})\|^2_{L^2}, \end{aligned}$$

where the second step follows from the elementary inequality $ab \leq a^2/2 + b^2/2$ and the non-expansivity of projection onto the space of non-negative functions. By the Pythagorean theorem, we have

$$\begin{aligned} \|m^t_{w\to v} - m^*_{w\to v}\|^2_{L^2} &= \|m^t_{w\to v} - \Pi^r(m^*_{w\to v})\|^2_{L^2} + \|\Pi^r(m^*_{w\to v}) - m^*_{w\to v}\|^2_{L^2} \\ &= \|\Delta^t_{w\to v}\|^2_{L^2} + \|A^r_{w\to v}\|^2_{L^2}. \end{aligned}$$

Using this equality and taking expectations, we obtain

$$\begin{aligned} \mathbb{E}[T] &\leq \big(1-\frac{\gamma}{2}\big)\Big\{\frac{1}{2}\,\frac{\sum_{w\in\mathcal{N}(v)\setminus\{u\}}[\bar{\rho}^2(\Delta^t_{w\to v}) + \|A^r_{w\to v}\|^2_{L^2}]}{|\mathcal{N}(v)|-1} + \frac{1}{2}\,\bar{\rho}^2(\Delta^t_{v\to u})\Big\} - \bar{\rho}^2(\Delta^t_{v\to u}) \\ &\leq \big(\frac{1}{2}-\frac{\gamma}{4}\big)\,\rho^2_{\max}(A^r) + \big(\frac{1}{2}-\frac{\gamma}{4}\big)\,\bar{\rho}^2_{\max}(\Delta^t) - \big(\frac{1}{2}+\frac{\gamma}{4}\big)\,\bar{\rho}^2(\Delta^t_{v\to u}). \end{aligned}$$

Since $V^t_{v\to u} = 2\eta^t\,\mathbb{E}[T]$, the claim follows.

References

A. Agarwal, P. L. Bartlett, P. Ravikumar, and M. J. Wainwright. Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. *IEEE Transactions on Information Theory*, 58(5):3235–3249, May 2012.

S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2):325–343, March 2000.

M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transaction on Signal Processing*, 50(2):174–188, 2002.

S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, March 2011.

G. Boccignone, A. Marcelli, P. Napoletano, and M. Ferraro. Motion estimation via belief propagation. In *Proceedings of the International Conference on Image Analysis and Processing*, pages 55–60, 2007.

F. Chung and L. Lu. Concentration inequalities and martingale inequalities: A survey. *Internet Mathematics*, 3(1):79–127, 2006.

F. Clarke. *Functional Analysis, Calculus of Variations and Optimal Control*. Springer-Verlag, London, 2013.

J. Coughlan and H. Shen. Dynamic quantization for belief propagation in sparse spaces. *Computer Vision and Image Understanding*, 106(1):47–58, 2007.

A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, New York, 2001.

R. Durrett. *Probability: Theory and Examples*. Duxbury Press, New York, NY, 1995.

M. Fabian, P. Habala, P. Hajek, V. Montesinos, and V. Zizler. *Banach Space Theory: The Basis for Linear and Nonlinear Analysis*. Springer, New York, 2011.

C. Gu. *Smoothing Spline ANOVA Models*. Springer Series in Statistics. Springer, New York, NY, 2002.

A. Ihler, J. Fisher, and A. S. Willsky. Loopy belief propagation: Convergence and effects of message errors. *Journal of Machine Learning Research*, 6:905–936, May 2005.

A. T. Ihler and D. McAllester. Particle belief propagation. In *Proceedings of the Conference on Artificial Intelligence and Statistics*, pages 256–263, 2009.

M. Isard. PAMPAS: Real-valued graphical models for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 613–620, 2003.

M. Isard, J. MacCormick, and K. Achan. Continuously-adaptive discretization for message-passing algorithms. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 737–744, 2009.

F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transaction on Information Theory*, 47(2):498–519, 2001.

H. A. Loeliger. An introduction to factor graphs. *IEEE Signal Processing Magazine*, 21(1):28–41, 2004.

D. G. Luenberger. *Optimization by Vector Space Methods*. Wiley, New York, 1969.

J. M. Mooij and H. J. Kappen. Sufficient conditions for convergence of the sum-product algorithm. *IEEE Transactions on Information Theory*, 53(12):4422–4437, December 2007.

A. S. Nemirovsky and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. New York, 1983.

N. Noorshams and M. J. Wainwright. Quantized stochastic belief propagation: Efficient message-passing for continuous state spaces. In *Proceedings of the IEEE International Symposium on Information Theory*, 2012.

N. Noorshams and M. J. Wainwright. Stochastic belief propagation: A low-complexity alternative to the sum-product algorithms. *IEEE Transactions on Information Theory*, 59(4):1981–2000, April 2013.

F. Riesz and B.S. Nagy. *Functional Analysis*. Dover Publications Inc., New York, 1990.

B. D. Ripley. *Stochastic Simulation*. Wiley, New York, 1987.

T. G. Roosta, M. J. Wainwright, and S. S. Sastry. Convergence analysis of reweighted sum-product algorithms. *IEEE Transactions on Signal Processing*, 56(9):4293–4305, September 2008.

H. L. Royden. *Real Analysis*. Prentice-Hall, New Jersey, 1988.

L. Song, A. Gretton, D. Bickson, Y. Low, and C. Guestrin. Kernel belief propagation. In *Proceedings of the Artificial Intelligence and Statistics*, 2011.

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

E. B. Sudderth, A. T. Ihler, M. Israd, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. *Communications of the ACM Magazine*, 53(10):95–103, 2010.

S. Tatikonda and M. I. Jordan. Loopy belief propagation and Gibbs measures. In *Proc. Uncertainty in Artificial Intelligence*, volume 18, pages 493–500, August 2002.

M. J. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc, Hanover, MA 02339, USA, 2008.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Transaction on Information Theory*, 51(7):2282–2312, July 2005.

Journal of Machine Learning Research 14 (2013) 2837-2856 Submitted 3/13; Revised 6/13; Published 9/13

A Binary-Classification-Based Metric between Time-Series Distributions and Its Use in Statistical and Learning Problems

Daniil Ryabko DANIIL.RYABKO@INRIA.FR
Jérémie Mary JEREMIE.MARY@INRIA.FR
SequeL-INRIA/LIFL-CNRS
Université de Lille, France
40, avenue de Halley
59650 Villeneuve d'Ascq
France

Editor: Léon Bottou

Abstract

A metric between time-series distributions is proposed that can be evaluated using binary classification methods, which were originally developed to work on i.i.d. data. It is shown how this metric can be used for solving statistical problems that are seemingly unrelated to classification and concern highly dependent time series. Specifically, the problems of time-series clustering, homogeneity testing and the three-sample problem are addressed. Universal consistency of the resulting algorithms is proven under most general assumptions. The theoretical results are illustrated with experiments on synthetic and real-world data.

Keywords: time series, reductions, stationary ergodic, clustering, metrics between probability distributions

1. Introduction

Binary classification is one of the most well-understood problems of machine learning and statistics: a wealth of efficient classification algorithms has been developed and applied to a wide range of applications. Perhaps one of the reasons for this is that binary classification is conceptually one of the simplest statistical learning problems. It is thus natural to try and use it as a building block for solving other, more complex, newer or just different problems; in other words, one can try to obtain efficient algorithms for different learning problems by reducing them to binary classification. This approach has been applied to many different problems, starting with multi-class classification, and including regression and ranking (Balcan et al., 2007; Langford et al., 2006), to give just a few examples. However, all of these problems are formulated in terms of independent and identically distributed (i.i.d.) samples. This is also the assumption underlying the theoretical analysis of most of the classification algorithms.

In this work we consider learning problems that concern time-series data for which independence assumptions do not hold. The series can exhibit arbitrary long-range dependence, and different time-series samples may be interdependent as well. Moreover, the learning problems that we consider—the three-sample problem, time-series clustering, and homogeneity testing—at first glance seem completely unrelated to classification.

We show how the considered problems can be reduced to binary classification methods, via a new metric between time-series distributions. The results include asymptotically consistent algorithms, as well as finite-sample analysis. To establish the consistency of the suggested methods, for clustering and the three-sample problem the only assumption that we make on the data is that the distributions generating the samples are stationary ergodic; this is one of the weakest assumptions used in statistics. For homogeneity testing we have to make some mixing assumptions in order to obtain consistency results (this is indeed unavoidable, as shown by Ryabko, 2010b). Mixing conditions are also used to obtain finite-sample performance guarantees for the first two problems.

The proposed approach is based on a new distance between time-series distributions (that is, between probability distributions on the space of infinite sequences), which we call *telescope distance*. This distance can be evaluated using binary classification methods, and its finite-sample estimates are shown to be asymptotically consistent. Three main building blocks are used to construct the telescope distance. The first one is a distance on finite-dimensional marginal distributions. The distance we use for this is the following well-known metric: $d_{\mathcal{H}}(P,Q) := \sup_{h\in\mathcal{H}} |\mathbf{E}_P h - \mathbf{E}_Q h|$ where P, Q are distributions and $\mathcal{H}$ is a set of functions. This distance can be estimated using binary classification methods, and thus can be used to reduce various statistical problems to the classification problem. This distance was previously applied to such statistical problems as homogeneity testing and change-point estimation (Kifer et al., 2004). However, these applications so far have only concerned i.i.d. data, whereas we want to work with highly-dependent time series. Thus, the second building block are the recent results of Adams and Nobel (2012), that show that empirical estimates of $d_{\mathcal{H}}$ are consistent (under certain conditions on $\mathcal{H}$) for arbitrary stationary ergodic distributions. This, however, is not enough: evaluating $d_{\mathcal{H}}$ for (stationary ergodic) time-series distributions means measuring the distance between their finite-dimensional marginals, and not the distributions themselves. Finally, the third step to construct the distance is what we call *telescoping*. It consists in summing the distances for all the (infinitely many) finite-dimensional marginals with decreasing weights. The resulting distance can "automatically" select the marginal distribution of the right order: marginals which cannot distinguish between the distributions give distance estimates that converge to zero, while marginals whose orders are too high to have converged have very small weights. Thus, the estimate is dominated by the marginals which can distinguish between the time-series distributions, or converges to zero if the distributions are the same. It is worth noting that a similar telescoping trick is used in different problems, most notably, in sequence prediction (Solomonoff, 1978; B. Ryabko, 1988; Ryabko, 2011); it is also used in the distributional distance (Gray, 1988), see Section 8 below.

We show that the resulting distance (telescope distance) indeed can be consistently estimated based on sampling, for arbitrary stationary ergodic distributions. Further, we show how this fact can be used to construct consistent algorithms for the considered problems on time series. Thus we can harness binary classification methods to solve statistical learning problems concerning time series. A remarkable feature of the resulting methods is that the performance guarantees obtained do not depend on the approximation error of the binary classification methods used, they only depend on their estimation error.

Moreover, we analyse some other distances between time-series distributions, the possibility of their use for solving the statistical problems considered, and the relation of these distances to the telescope distance introduced in this work.

To illustrate the theoretical results in an experimental setting, we chose the problem of time-series clustering, since it is a difficult unsupervised problem which seems most different from the

problem of binary classification. Experiments on both synthetic and real-world data are provided. The real-world setting concerns brain-computer interface (BCI) data, which is a notoriously challenging application, and on which the presented algorithm demonstrates competitive performance.

A related approach to address the problems considered here, as well as some related problems about stationary ergodic time series, is based on (consistent) empirical estimates of the distributional distance, see Ryabko and Ryabko (2010), Ryabko (2010a), Khaleghi et al. (2012), as well as Gray (1988) about the distributional distance. The empirical distance is based on counting frequencies of bins of decreasing sizes and "telescoping." This distance is described in some detail in Section 8 below, where we compare it to the telescope distance. Another related approach to time-series analysis involves a different reduction, namely, that to data compression (B. Ryabko, 2009).

1.1 Organisation

Section 2 is preliminary. In Section 3 we introduce and discuss the telescope distance. Section 4 explains how this distance can be calculated using binary classification methods. Sections 5 and 6 are devoted to the three-sample problem and clustering, respectively. In Section 7, under some mixing conditions, we address the problems of homogeneity testing, clustering with unknown k, and finite-sample performance guarantees. In Section 8 we take a look at other distances between time-series distributions and their relations to the telescope distance. Section 9 presents experimental evaluation.

2. Notation and Definitions

Let $(\mathcal{X}, \mathcal{F}_1)$ be a measurable space (the domain), and denote $(\mathcal{X}^k, \mathcal{F}_k)$ and $(\mathcal{X}^{\mathbb{N}}, \mathcal{F})$ the product probability space over $\mathcal{X}^k$ and the induced probability space over the one-way infinite sequences taking values in $\mathcal{X}$. Time-series (or process) distributions are probability measures on the space $(X^{\mathbb{N}}, \mathcal{F})$. We use the abbreviation $X_{1..k}$ for $X_1, \dots, X_k$. A set $\mathcal{H}$ of functions is called *separable* if there is a countable set $\mathcal{H}'$ of functions such that any function in $\mathcal{H}$ is a pointwise limit of a sequence of elements of $\mathcal{H}'$.

A distribution ρ is called stationary if $\rho(X_{1..k} \in A) = \rho(X_{n+1..n+k} \in A)$ for all $A \in \mathcal{F}_k$, $k, n \in \mathbb{N}$. A stationary distribution is called (stationary) ergodic if

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1..n-k+1} \mathbb{I}_{X_{i..i+k} \in A} = \rho(A) \;\; \rho - \text{a.s.}$$

for every $A \in \mathcal{F}_k$, $k \in \mathbb{N}$. (This definition, which is more suited for the purposes of this work, is equivalent to the usual one expressed in terms of invariant sets, see, e.g., Gray, 1988.)

3. A Distance between Time-Series Distributions

We start with a distance between distributions on $\mathcal{X}$, and then we extend it to distributions on $\mathcal{X}^{\mathbb{N}}$. For two probability distributions P and Q on $(\mathcal{X}, \mathcal{F}_1)$ and a set $\mathcal{H}$ of measurable functions on $\mathcal{X}$, one can define the distance

$$d_{\mathcal{H}}(P, Q) := \sup_{h \in \mathcal{H}} |\mathbf{E}_P h - \mathbf{E}_Q h|. \tag{1}$$

This metric in its general form has been studied at least since the 80's (Zolotarev, 1983); its special cases include Kolmogorov-Smirnov (Kolmogorov, 1933), Kantorovich-Rubinstein (Kantorovich

and Rubinstein, 1957) and Fortet-Mourier (Fortet and Mourier, 1953) metrics. Note that the distance function so defined may not be measurable; however, it is measurable under mild conditions which we assume whenever necessary. In particular, separability of $\mathcal{H}$ is a sufficient condition (separability is required in most of the results below).

We are interested in the cases where $d_{\mathcal{H}}(P,Q)=0$ implies $P=Q$. Note that in this case $d_{\mathcal{H}}$ is a metric (the rest of the properties are easy to see). For reasons that will become apparent shortly (see Remark below), we are mainly interested in the sets $\mathcal{H}$ that consist of indicator functions. In this case we can identify each $f\in\mathcal{H}$ with the indicator set $\{x: f(x)=1\}\subset\mathcal{X}$ and (by a slight abuse of notation) write $d_{\mathcal{H}}(P,Q):=\sup_{h\in\mathcal{H}}|P(h)-Q(h)|$. In this case it is easy to check that the following statement holds true.

Lemma 1 *$d_{\mathcal{H}}$ is a metric on the space of probability distributions over $\mathcal{X}$ if and only if $\mathcal{H}$ generates $\mathcal{F}_1$.*

The property that $\mathcal{H}$ generates $\mathcal{F}_1$ is often easy to verify directly. First of all, it trivially holds for the case where $\mathcal{H}$ is the set of halfspaces in a Euclidean $\mathcal{X}$. It is also easy to check that it holds if $\mathcal{H}$ is the set of halfspaces in the feature space of most commonly used kernels (provided the feature space is of the same or higher dimension than the input space), such as polynomial and Gaussian kernels.

Based on $d_{\mathcal{H}}$ we can construct a distance between time-series probability distributions. For two time-series distributions ρ_1,ρ_2 we take the $d_{\mathcal{H}}$ between k-dimensional marginal distributions of ρ_1 and ρ_2 for each $k\in\mathbb{N}$, and sum them all up with decreasing weights.

Definition 2 (telescope distance $D_{\mathbf{H}}$) *For two time series distributions ρ_1 and ρ_2 on the space $(\mathcal{X}^{\mathbb{N}},\mathcal{F})$ and a sequence of sets of functions $\mathbf{H}=(\mathcal{H}_1,\mathcal{H}_2,\dots)$ define the* telescope distance

$$D_{\mathbf{H}}(\rho_1,\rho_2):=\sum_{k=1}^{\infty} w_k \sup_{h\in\mathcal{H}_k}|\mathbf{E}_{\rho_1}h(X_1,\dots,X_k)-\mathbf{E}_{\rho_2}h(Y_1,\dots,Y_k)|, \tag{2}$$

where w_k, $k\in\mathbb{N}$ is a sequence of positive summable real weights (e.g., $w_k=1/k^2$ or $w_k=2^{-k}$).

Lemma 3 *$D_{\mathbf{H}}$ is a metric if and only if $d_{\mathcal{H}_k}$ is a metric for every $k\in\mathbb{N}$.*

Proof The statement follows from the fact that two process distributions are the same if and only if all their finite-dimensional marginals coincide. ■

Definition 4 (empirical telescope distance $\hat{D}$) *For a pair of samples $X_{1..n}$ and $Y_{1..m}$ define the* empirical telescope distance *as*

$$\hat{D}_{\mathbf{H}}(X_{1..n},Y_{1..m}):= \sum_{k=1}^{\min\{m,n\}} w_k \sup_{h\in\mathcal{H}_k}\left|\frac{1}{n-k+1}\sum_{i=1}^{n-k+1}h(X_{i..i+k-1})-\frac{1}{m-k+1}\sum_{i=1}^{m-k+1}h(Y_{i..i+k-1})\right|. \tag{3}$$

All the methods presented in this work are based on the empirical telescope distance. The key fact is that it is an asymptotically consistent estimate of the telescope distance, that is, the latter can be consistently estimated based on sampling.

Theorem 5 *Let $\mathbf{H} = (\mathcal{H}_k)_{k\in\mathbb{N}}$ be a sequence of separable sets $\mathcal{H}_k$ of indicator functions (over X^k) of finite VC dimension such that $\mathcal{H}_k$ generates $\mathcal{F}_k$. Then for every stationary ergodic time series distributions ρ_X and ρ_Y generating samples $X_{1..n}$ and $Y_{1..m}$ we have*

$$\lim_{n,m\to\infty} \hat{D}_{\mathbf{H}}(X_{1..n}, Y_{1..m}) = D_{\mathbf{H}}(\rho_X, \rho_Y) a.s.$$

Note that $\hat{D}_{\mathbf{H}}$ is a biased estimate of $D_{\mathbf{H}}$, and, unlike in the i.i.d. case, the bias may depend on the distributions; however, the bias is $o(n)$.

Remark. The condition that the sets $\mathcal{H}_k$ are sets of indicator function of finite VC dimension comes from the results of Adams and Nobel (2012), who show that for any stationary ergodic distribution ρ, under these conditions, $\sup_{h\in\mathcal{H}_k} \frac{1}{n-k+1}\sum_{i=1}^{n-k+1} h(X_{i..i+k-1})$ is an asymptotically consistent estimate of $\sup_{h\in\mathcal{H}_k} \mathbf{E}_\rho h(X_1,\dots,X_k)$. This fact implies that $d_{\mathcal{H}_k}$ can be consistently estimated, from which the theorem is derived.

Proof [of Theorem 5] As established by Adams and Nobel (2012), under the conditions of the theorem we have

$$\lim_{n\to\infty} \sup_{h\in\mathcal{H}_k} \frac{1}{n-k+1} \sum_{i=1}^{n-k+1} h(X_{i..i+k-1}) = \sup_{h\in\mathcal{H}_k} \mathbf{E}_{\rho_X} h(X_1,\dots,X_k)\ \rho_X\text{-a.s.} \tag{4}$$

for all $k\in\mathbb{N}$, and likewise for ρ_Y. Fix an $\varepsilon > 0$. We can find a $T\in\mathbb{N}$ such that

$$\sum_{k>T} w_k \le \varepsilon. \tag{5}$$

Note that T depends only on ε. Moreover, as follows from (4), for each $k = 1..T$ we can find an N_k such that

$$\left| \sup_{h\in\mathcal{H}_k} \frac{1}{n-k+1} \sum_{i=1}^{n-k+1} h(X_{i..i+k-1}) - \sup_{h\in\mathcal{H}_k} \mathbf{E}_{\rho_X} h(X_{1..k}) \right| \le \varepsilon/T. \tag{6}$$

Let $N_k := \max_{i=1..T} N_i$ and define analogously M for ρ_Y. Thus, for $n\ge N$, $m\ge M$ we have

$$\begin{aligned}
&\hat{D}_{\mathbf{H}}(X_{1..n}, Y_{1..m}) \\
&\quad\le \sum_{k=1}^{T} w_k \sup_{h\in\mathcal{H}_k} \left| \frac{1}{n-k+1} \sum_{i=1}^{n-k+1} h(X_{i..i+k-1}) - \frac{1}{m-k+1} \sum_{i=1}^{m-k+1} h(Y_{i..i+k-1}) \right| + \varepsilon \\
&\quad\le \sum_{k=1}^{T} w_k \sup_{h\in\mathcal{H}_k} \left\{ \left| \frac{1}{n-k+1} \sum_{i=1}^{n-k+1} h(X_{i..i+k-1}) - \mathbf{E}_{\rho_1} h(X_{1..k}) \right| \right. \\
&\qquad\qquad + |\mathbf{E}_{\rho_1} h(X_{1..k}) - \mathbf{E}_{\rho_2} h(Y_{1..k})| \\
&\qquad \left. + \left| \mathbf{E}_{\rho_2} h(Y_{1..k}) - \frac{1}{m-k+1} \sum_{i=1}^{m-k+1} h(Y_{i..i+k-1}) \right| \right\} + \varepsilon \\
&\qquad\qquad\qquad\qquad \le 3\varepsilon + D_{\mathbf{H}}(\rho_X, \rho_Y),
\end{aligned}$$

where the first inequality follows from the definition (3) of $\hat{D}_{\mathbf{H}}$ and from (5), and the last inequality follows from (6). Since ε was chosen arbitrary the statement follows. ■

4. Calculating $\hat{D}_{\mathbf{H}}$ Using Binary-Classification Methods

The methods for solving various statistical problems that we suggest are all based on $\hat{D}_{\mathbf{H}}$. The main appeal of this approach is that $\hat{D}_{\mathbf{H}}$ can be calculated using binary classification methods. Here we explain how to do it.

The definition (3) of $D_{\mathbf{H}}$ involves calculating l summands (where $l := \min\{n,m\}$), that is

$$\sup_{h\in\mathcal{H}_k}\left|\frac{1}{n-k+1}\sum_{i=1}^{n-k+1}h(X_{i..i+k-1})-\frac{1}{m-k+1}\sum_{i=1}^{m-k+1}h(Y_{i..i+k-1})\right| \tag{7}$$

for each $k = 1..l$. Assuming that $h \in \mathcal{H}_k$ are indicator functions, calculating each of the summands amounts to solving the following k-dimensional binary classification problem. Consider $X_{i..i+k-1}$, $i = 1..n-k+1$ as class-1 examples and $Y_{i..i+k-1}$, $i = 1..m-k+1$ as class-0 examples. The supremum (7) is attained on $h \in \mathcal{H}_k$ that minimizes the empirical risk, with examples weighted with respect to the sample size. Indeed, we can define the weighted empirical risk of any $h \in \mathcal{H}_k$ as

$$\frac{1}{n-k+1}\sum_{i=1}^{n-k+1}(1-h(X_{i..i+k-1}))+\frac{1}{m-k+1}\sum_{i=1}^{m-k+1}h(Y_{i..i+k-1}), \tag{8}$$

minimising which can be easily seen to be equivalent to (7).

Thus, as long as we have a way to find $h \in \mathcal{H}_k$ that minimizes empirical risk, we have a consistent estimate of $D_{\mathcal{H}}(\rho_X,\rho_Y)$, under the mild conditions on $\mathbf{H}$ required by Theorem 5. Since the dimension of the resulting classification problems grows with the length of the sequences, one should prefer methods that work in high dimensions, such as soft-margin SVMs (Cortes and Vapnik, 1995).

A particularly remarkable feature is that *the choice of $\mathcal{H}_k$ is much easier* for the problems that we consider *than in the binary classification* problem. Specifically, if (for some fixed k) the classifier that achieves the minimal (Bayes) error for the classification problem is not in $\mathcal{H}_k$, then obviously the error of an empirical risk minimizer will not tend to zero, no matter how much data we have. In contrast, all we need to achieve asymptotically 0 error in estimating $\hat{D}$ (and therefore, in the learning problems considered below) is that the sets $\mathcal{H}_k$ generate $\mathcal{F}_k$ and have a finite VC dimension (for each k). This is the case already for the set of half-spaces in $\mathbb{R}_k$. In other words, the *approximation* error of the binary classification method (the classification error of the best f in $\mathcal{H}_k$) is not important. What is important is the estimation error; for asymptotic consistency results it has to go to 0 (hence the requirement on the VC dimension); for non-asymptotic results, it will appear in the error bounds, see Section 7. Thus, we have the following statement.

Claim 1 *The error $|D_{\mathbf{H}}(\rho_X,\rho_Y)-\hat{D}_{\mathbf{H}}(X,Y)|$, and thus the error of the algorithms below, can be much smaller than the error of classification algorithms used to calculate $D_{\mathbf{H}}(X,Y)$.*

We can conclude that, beyond the requirement that $\mathcal{H}_k$ generate $\mathcal{F}_k$ for each $k \in \mathbb{N}$, the choice of H_k (or, say, of the kernel to use in SVM) is entirely up to the needs and constraints of specific applications.

Remark (number of summands in $\hat{D}_{\mathbf{H}}$) Finally, we note that while in the definition of the empirical distributional distance (3) the number of summands is l (the length of the shorter of the two

samples), it can be replaced with any γ_l such that $\gamma_l \to \infty$, without affecting any asymptotic consistency results. In other words, Theorem 5, as well as all the consistency statements below, holds true for l replaced with any non-decreasing function γ_l that tends to infinity with l. A practically viable choice is $\gamma_l = \log l$; in fact, there is no reason to choose faster growing γ_n since the estimates for higher-order summands will not have enough data to converge. This is also the value we use in the experiments.

Remark (relation to total variation) An illustrative example[1] of the choice of the sets $\mathcal{H}_k$ is the set of indicators of all measurable subsets of $\mathcal{X}^k$. In this case each summand in (2) is the total variation distance between the k-dimensional marginal distributions of ρ_1 and ρ_2. Take, for simplicity, $k = 1$; denoting P and Q the corresponding single-dimensional marginals, the distance becomes $\sup_A |P(A) - Q(A)|$ (cf. (1)). This supremum is reached on the set $A^* := \{x \in \mathcal{X} : f(x) \geq g(x)\}$, where f and g are densities of P and Q with respect to some arbitrary measure that dominates both P and Q (e.g., $1/2(P+Q)$). A binary classifier corresponding to a set A declares P if $x \in A$ and Q otherwise. The optimal classification error is $\inf_A(1 - P(A) + Q(A)) = 1 - \sup_A(P(A) + Q(A)) = 1 - P(A^*) + Q(A^*)$ (cf. (8)). In general, estimating the total variation distance (and finding the best classifier) is not possible, so using smaller sets $\mathcal{H}_k$ can be viewed as a regularization of this problem.

5. The Three-Sample Problem

We start with a conceptually simple problem known in statistics as the three-sample problem (sometimes also called time-series classification). We are given three samples $X = (X_1, \ldots, X_n)$, $Y = (Y_1, \ldots, Y_m)$ and $Z = (Z_1, \ldots, Z_l)$. It is known that X and Y were generated by different time-series distributions, whereas Z was generated by the same distribution as either X or Y. It is required to find out which one is the case. Both distributions are assumed to be stationary ergodic, but no further assumptions are made about them (no independence, mixing or memory assumptions). The three sample-problem for dependent time series has been addressed by Gutman (1989) for Markov processes and by Ryabko and Ryabko (2010) for stationary ergodic time series. The latter work uses an approach based on the distributional distance.

Indeed, to solve this problem it suffices to have consistent estimates of some distance between time series distributions. Thus, we can use the telescope distance. The following statement is a simple corollary of Theorem 5.

Theorem 6 *Let the samples $X = (X_1, \ldots, X_n)$, $Y = (Y_1, \ldots, Y_m)$ and $Z = (Z_1, \ldots, Z_l)$ be generated by stationary ergodic distributions ρ_X, ρ_Y and ρ_Z, with $\rho_X \neq \rho_Y$ and either (i) $\rho_Z = \rho_X$ or (ii) $\rho_Z = \rho_Y$. Let the sets $\mathcal{H}_k$, $k \in \mathbb{N}$ be separable sets of indicator functions over $\mathcal{X}^k$. Assume that each set $\mathcal{H}_k$, $k \in \mathbb{N}$ has a finite VC dimension and generates $\mathcal{F}_k$. A test that declares that (i) is true if $\hat{D}_{\mathbf{H}}(Z,X) \leq \hat{D}_{\mathbf{H}}(Z,Y)$ and that (ii) is true otherwise, makes only finitely many errors with probability 1 as $n, m, l \to \infty$.*

It is straightforward to extend this theorem to more than two classes; in other words, instead of X and Y one can have an arbitrary number of samples from different stationary ergodic distributions. A further generalization of this problem is the problem of time-series clustering, considered in the next section.

1. This example was suggested by an anonymous reviewer.

6. Clustering Time Series

We are given N time-series samples $X^1 = (X_1^1, \dots, X_{n_1}^1), \dots, X^N = (X_1^N, \dots, X_{n_N}^N)$, and it is required to cluster them into K groups, where, in different settings, K may be either known or unknown. While there may be many different approaches to define what should be considered a good clustering, and, thus, what it means to have a consistent clustering algorithm, for the problem of clustering time-series samples there is a natural choice, proposed by Ryabko (2010a): Assume that each of the time-series samples $X^1 = (X_1^1, \dots, X_{n_1}^1), \dots, X^N = (X_1^N, \dots, X_{n_N}^N)$ was generated by one out of K different time-series distributions $\rho_1, \dots, \rho_K$. These distributions are unknown. The *target clustering* is defined according to whether the samples were generated by the same or different distributions: the samples belong to the same cluster if and only if they were generated by the same distribution. A clustering algorithm is called *asymptotically consistent* if with probability 1 from some n on it outputs the target clustering, where n is the length of the shortest sample $n := \min_{i=1..N} n_i \geq n'$.

Again, to solve this problem it is enough to have a metric between time-series distributions that can be consistently estimated. Our approach here is based on the telescope distance, and thus we use $\hat{D}$.

The clustering problem is relatively simple if the target clustering has what is called the *strict separation property* (Balcan et al., 2008): every two points in the same target cluster are closer to each other than to any point from a different target cluster. The following statement is an easy corollary of Theorem 5.

Theorem 7 *Let the sets $\mathcal{H}_k$, $k \in \mathbb{N}$ be separable sets of indicator functions over $\mathcal{X}^k$. Assume that each set $\mathcal{H}_k$, $k \in \mathbb{N}$ has a finite VC dimension and generates $\mathcal{F}_k$. If the distributions $\rho_1, \dots, \rho_K$ generating the samples $X^1 = (X_1^1, \dots, X_{n_1}^1), \dots, X^N = (X_1^N, \dots, X_{n_N}^N)$ are stationary ergodic, then with probability 1 from some $n := \min_{i=1..N} n_i$ on the target clustering has the strict separation property with respect to $\hat{D}_{\mathbf{H}}$.*

With the strict separation property at hand, if the number of clusters K is known, it is easy to find asymptotically consistent algorithms. Here we give some simple examples, but the theorem below can be extended to many other distance-based clustering algorithms.

The *average linkage* algorithm works as follows. The distance between clusters is defined as the average distance between points in these clusters. First, put each point into a separate cluster. Then, merge the two closest clusters; repeat the last step until the total number of clusters is K. The *farthest point* clustering works as follows. Assign $c_1 := X^1$ to the first cluster. For $i = 2..K$, find the point X^j, $j \in \{1..N\}$ that maximizes the distance $\min_{t=1..i} \hat{D}_{\mathbf{H}}(X^j, c_t)$ (to the points already assigned to clusters) and assign $c_i := X^j$ to the cluster i. Then assign each of the remaining points to the nearest cluster. The following statement is a corollary of Theorem 7.

Theorem 8 *Under the conditions of Theorem 7, average linkage and farthest point clusterings are asymptotically consistent, provided the correct number of clusters K is given to the algorithm.*

Note that we do not require the samples to be independent; the joint distributions of the samples may be completely arbitrary, as long as the marginal distribution of each sample is stationary ergodic. These results can be extended to the online setting in the spirit of Khaleghi et al. (2012).

For the case of unknown number of clusters, the situation is different: one has to make stronger assumptions on the distributions generating the samples, since there is no algorithm that is consistent for all stationary ergodic distributions (Ryabko, 2010b); such stronger assumptions are considered in the next section.

7. Speed of Convergence

The results established so far are asymptotic out of necessity: they are established under the assumption that the distributions involved are stationary ergodic, which is too general to allow for any meaningful finite-time performance guarantees. While it is interesting to be able to establish consistency results under such general assumptions, it is also interesting to see what results can be obtained under stronger assumptions. Moreover, since it is usually not known in advance whether the data at hand satisfies given assumptions or not, it appears important to have methods that have *both* asymptotic consistency in the general setting and finite-time performance guarantees under stronger assumptions. It turns out that this is possible: for the methods based on $\hat{D}$ one can establish both the asymptotic performance guarantees for all stationary ergodic distributions and finite-sample performance guarantees under stronger assumptions, namely the uniform mixing conditions introduced below.

Another reason to consider stronger assumptions on the distributions generating the data is that some statistical problems, such as homogeneity testing or clustering when the number of clusters is unknown, are provably impossible to solve under the only assumption of stationary ergodic distributions, as shown by Ryabko (2010b).

Thus, in this section we analyse the speed of convergence of $\hat{D}$ under certain mixing conditions, and use it to construct solutions for the problems of homogeneity and clustering with an unknown number of clusters, as well as to establish finite-time performance guarantees for the methods presented in the previous sections.

A stationary distribution on the space of one-way infinite sequences $(\mathcal{X}^{\mathbb{N}}, \mathcal{F})$ can be uniquely extended to a stationary distribution on the space of two-way infinite sequences $(\mathcal{X}^{\mathbb{Z}}, \mathcal{F}_{\mathbb{Z}})$ of the form $\ldots, X_{-1}, X_0, X_1, \ldots$.

Definition 9 (β-mixing coefficients) *For a process distribution* ρ *define the mixing coefficients*

$$\beta(\rho, k) := \sup_{\substack{A \in \sigma(X_{-\infty..0}),\\ B \in \sigma(X_{k..\infty})}} |\rho(A \cap B) - \rho(A)\rho(B)|$$

where $\sigma(..)$ *denotes the sigma-algebra of the random variables in brackets.*

When $\beta(\rho, k) \to 0$ the process ρ is called uniformly β-mixing (with coefficients $\beta(\rho, k)$); this condition is much stronger than ergodicity, but is much weaker than the i.i.d. assumption. For more information on mixing see, for example, Bosq (1996).

7.1 Speed of Convergence of $\hat{D}$

Assume that a sample $X_{1..n}$ is generated by a distribution ρ that is uniformly β-mixing with coefficients $\beta(\rho, k)$. Assume further that $\mathcal{H}_k$ is a set of indicator functions with a finite VC dimension d_k, for each $k \in \mathbb{N}$.

Since in this section we are after finite-time bounds, we fix a concrete choice of the weights w_k in the definition of $\hat{D}$ (Definition 2),

$$w_k := 2^{-k}. \tag{9}$$

The general tool that we use to obtain performance guarantees in this section is the following bound that can be obtained from the results of Karandikar and Vidyasagar (2002).

$$q_n(\rho,\mathcal{H}_k,\varepsilon) := \rho\left(\sup_{h\in\mathcal{H}_k}\left|\frac{1}{n-k+1}\sum_{i=1}^{n-k+1}h(X_{i..i+k-1})-\mathbf{E}_\rho h(X_{1..k})\right|>\varepsilon\right)$$
$$\le n\beta(\rho,t_n-k)+8t_n^{d_k+1}e^{-l_n\varepsilon^2/8}, \quad (10)$$

where t_n are any integers in $1..n$ and $l_n = n/t_n$. The parameters t_n should be set according to the values of β in order to optimize the bound.

One can use similar bounds for classes of finite Pollard dimension (Pollard, 1984) or more general bounds expressed in terms of covering numbers, such as those given by Karandikar and Vidyasagar (2002). Here we consider classes of finite VC dimension only for the ease of the exposition and for the sake of continuity with the previous section (where it was necessary).

Furthermore, for the rest of this section we assume geometric β-mixing distributions, that is, $\beta(\rho,t)\le\gamma^t$ for some $\gamma<1$. Letting $l_n = t_n = \sqrt{n}$ the bound (10) becomes

$$q_n(\rho,\mathcal{H}_k,\varepsilon)\le n\gamma^{\sqrt{n}-k}+8n^{(d_k+1)/2}e^{-\sqrt{n}\varepsilon^2/8}. \quad (11)$$

Lemma 10 *Let two samples $X_{1..n}$ and $Y_{1..m}$ be generated by stationary distributions ρ_X and ρ_Y whose β-mixing coefficients satisfy $\beta(\rho_.,t)\le\gamma^t$ for some $\gamma<1$. Let $\mathcal{H}_k$, $k\in\mathbb{N}$ be some sets of indicator functions on $\mathcal{X}^k$ whose VC dimension d_k is finite and non-decreasing with k. Then*

$$P(|\hat{D}_{\mathbf{H}}(X_{1..n},Y_{1..m})-D_{\mathbf{H}}(\rho_X,\rho_Y)|>\varepsilon)\le 2\Delta(\varepsilon/4,n') \quad (12)$$

where $n' := \min\{n,m\}$, the probability is with respect to $\rho_X\times\rho_Y$ and

$$\Delta(\varepsilon,n) := -\log\varepsilon(n\gamma^{\sqrt{n}+\log(\varepsilon)}+8n^{(d_{-\log\varepsilon}+1)/2}e^{-\sqrt{n}\varepsilon^2/8}). \quad (13)$$

Proof From (9) we have $\sum_{k=-\log\varepsilon/2}^{\infty}w_k<\varepsilon/2$. Using this and the definitions 2 and 4 of $D_{\mathbf{H}}$ and $\hat{D}_{\mathbf{H}}$ we obtain

$$P(|\hat{D}_{\mathbf{H}}(X_{1..n_1},Y_{1..n_2})-D_{\mathbf{H}}(\rho_X,\rho_Y)|>\varepsilon)\le\sum_{k=1}^{-\log(\varepsilon/2)}(q_n(\rho_X,\mathcal{H}_k,\varepsilon/4)+q_n(\rho_Y,\mathcal{H}_k,\varepsilon/4)),$$

which, together with (11), implies the statement. ■

7.2 Homogeneity Testing

Given two samples $X_{1..n}$ and $Y_{1..m}$ generated by distributions ρ_X and ρ_Y respectively, the problem of homogeneity testing (or the two-sample problem) consists in deciding whether $\rho_X=\rho_Y$. A test is called (asymptotically) consistent if its probability of error goes to zero as $n' := \min\{m,n\}$ goes to infinity. As mentioned above, in general, for stationary ergodic time series distributions there is no asymptotically consistent test for homogeneity (Ryabko, 2010b) (even for binary-valued time series); thus, stronger assumptions are in order.

Homogeneity testing is one of the classical problems of mathematical statistics, and one of the most studied ones. Vast literature exits on homogeneity testing for i.i.d. data, and for dependent

processes as well. We do not attempt to survey this literature here. Our contribution to this line of research is to show that this problem can be reduced (via the telescope distance) to binary classification, in the case of strongly dependent processes satisfying some mixing conditions.

It is easy to see that under the mixing conditions of Lemma 10 a consistent test for homogeneity exists, and finite-sample performance guarantees can be obtained. It is enough to find a sequence $\varepsilon_n \to 0$ such that $\Delta(\varepsilon_n, n) \to 0$ (see (13)). Then the test can be constructed as follows: say that the two sequences $X_{1..n}$ and $Y_{1..m}$ were generated by the same distribution if $\hat{D}_{\mathbf{H}}(X_{1..n}, Y_{1..m}) < \varepsilon_{\min\{n,m\}}$; otherwise say that they were generated by different distributions.

Theorem 11 *Under the conditions of Lemma 10 the probability of Type I error (the distributions are the same but the test says they are different) of the described test is upper-bounded by* $2\Delta(\varepsilon/4, n')$. *The probability of Type II error (the distributions are different but the test says they are the same) is upper-bounded by* $2\Delta((\delta-\varepsilon)/4, n')$ *where* $\delta := D_{\mathbf{H}}(\rho_X, \rho_Y)$.

Proof The statement is an immediate consequence of Lemma 10. Indeed, for the Type I error, the two sequences are generated by the same distribution, so the probability of error of the test is given by (12) with $D_{\mathbf{H}}(\rho_X, \rho_Y) = 0$. The probability of Type II error is given by $P(D_{\mathbf{H}}(\rho_X, \rho_Y) - \hat{D}_{\mathbf{H}}(X_{1..n_1}, Y_{1..n_2}) > \delta - \varepsilon)$, which is upper-bounded by $2\Delta((\delta-\varepsilon))/4, n')$ as follows from (12). ■

The optimal choice of ε_n may depend on the speed at which d_k (the VC dimension of $\mathcal{H}_k$) increases; however, for most natural cases (recall that $\mathcal{H}_k$ are also parameters of the algorithm) this growth is polynomial, so the main term to control is $e^{-\sqrt{n}\varepsilon^2/8}$.

For example, if $\mathcal{H}_k$ is the set of halfspaces in $\mathcal{X}^k = \mathbb{R}^k$ then $d_k = k+1$ and one can choose $\varepsilon_n := n^{-1/8}$. The resulting probability of Type I error decreases as $\exp(-n^{1/4})$.

7.3 Clustering with a Known or Unknown Number of Clusters

If the distributions generating the samples satisfy certain mixing conditions, then we can augment Theorems 7 and 8 with finite-sample performance guarantees.

Theorem 12 *Let the distributions* $\rho_1, \dots, \rho_k$ *generating the samples* $X^1 = (X^1_1, \dots, X^1_{n_1}), \dots, X^N = (X^N_1, \dots, X^N_{n_N})$ *satisfy the conditions of Lemma 10. Let* $n := \min_{i=1..N} n_i$ *and* $\delta := \min_{i,j=1..N, i\neq j} D_{\mathbf{H}}(\rho_i, \rho_j)$. *Then with probability at least* $1 - N(N-1)\Delta(\delta/12, n')$ *the target clustering of the samples has the strict separation property. In this case single linkage and farthest point algorithms output the target clustering.*

Proof Note that a sufficient condition for the strict separation property to hold is that for every pair i, j of samples generated by the same distribution we have $\hat{D}_{\mathbf{H}}(X^i, X^j) \leq \delta/3$, and for every pair i, j of samples generated by different distributions we have $\hat{D}_{\mathbf{H}}(X^i, X^j) \geq 2\delta/3$. Using Lemma 10, the probability of such an even (for each pair) is upper-bounded by $2\Delta(\delta/12, n')$, which, multiplied by the total number $N(N-1)/2$ of pairs gives the statement. The second statement is obvious. ■

As with homogeneity testing, while in the general case of stationary ergodic distributions it is impossible to have a consistent clustering algorithm when the number of clusters k is unknown, the situation changes if the distributions satisfy certain mixing conditions. In this case a consistent clustering algorithm can be obtained as follows. Assign to the same cluster all samples that are at

most ε_n-far from each other, where the threshold ε_n is selected the same way as for homogeneity testing: $\varepsilon_n \to 0$ and $\Delta(\varepsilon_n, n) \to 0$. The optimal choice of this parameter depends on the choice of $\mathcal{H}_k$ through the speed of growth of the VC dimension d_k of these sets.

Theorem 13 *Given N samples generated by k different stationary distributions ρ_i, $i = 1..k$ (unknown k) all satisfying the conditions of Lemma 10, the probability of error (misclustering at least one sample) of the described algorithm is upper-bounded by*

$$N(N-1)\max\{\Delta(\varepsilon/4, n'), \Delta((\delta-\varepsilon)/4, n')\}$$

where $\delta := \min_{i,j=1..k, i\neq j} D_{\mathbf{H}}(\rho_i, \rho_j)$ and $n = \min_{i=1..N} n_i$, with n_i, $i = 1..N$ being lengths of the samples.

Proof The statement follows from Theorem 11. ■

8. Other Metrics for Time-Series Distributions

The previous sections introduce a new metric on the space of time-series distributions, and use its empirical estimates to solve several learning problems. In this section we attempt to put the telescope distance into a more general context, and take a broader look at metrics between time-series distributions.

Introduce the notation μ_k for the k-dimensional marginal distribution of a time-series distribution μ.

8.1 sum Distances

Observe that the telescope distance $D_{\mathbf{H}}$ has the form

$$D(\mu, \nu) = \sum_{k\in\mathbb{N}} w_k d_k(\mu_k, \nu_k), \tag{14}$$

where w_k are summable positive real weights.

It is easy to see that distances of this form can be consistently estimated, as long as d_k can be consistently estimated for each $k \in \mathbb{N}$; this is formalized in the following statement.

Proposition 14 (estimating sum-based distances) *Let $\mathcal{C}$ be a set of distributions over $\mathcal{X}^{\mathbb{N}}$. Let $d_k, k \in \mathbb{N}$ be a series of distances on the spaces of distributions over $\mathcal{X}^k$, such that $d_k(\mu_k, \nu_k) \leq a \in \mathbb{R}$ for all $\mu, \nu \in \mathcal{C}$ and such that there exists a series $\hat{d}_k(X_{1..n}, Y_{1..n}), k \in \mathbb{N}$ of their consistent estimates: for each $\mu, \nu \in \mathcal{C}$ we have $\lim_{n\to\infty} \hat{d}_k(X_{1..n}, Y_{1..n}) = d_k(\mu_k, \nu_k)$ a.s., whenever $\mu, \nu \in \mathcal{C}$ are chosen to generate the sequences. Then the distance D given by (14) can be consistently estimated using the estimate $\sum_{k\in\mathbb{N}} w_k \hat{d}_k(X_{1..n}, Y_{1..n})$.*

Proof The proof is an easy generalization of the proof of Theorem 5, with the condition on $\hat{d}_k$ used instead of (4). ■

Clearly, $D_{\mathbf{H}}$ is an example of a distance in the form (14), and it satisfies the conditions of the proposition with $\mathcal{C}$ being the set of all stationary ergodic processes.

Another example of a distance in the form (14) is given by the so-called distributional distance (Gray, 1988; Shields, 1996), whose definition is given below. Empirical estimates of this distance are asymptotically consistent for stationary ergodic time series, and thus can be used (Ryabko and Ryabko, 2010; Ryabko, 2010a; Khaleghi et al., 2012; Khaleghi and Ryabko, 2012; Ryabko, 2012) to solve various statistical problems, including those considered above.

To define the distributional distance, let, for each $k,l \in \mathbb{N}$, the set $B^{k,l}$ be some partition of the set $\mathcal{X}^k$, such that the set $B^k = \cup_{l\in\mathbb{N}} B^{k,l}$ generates $\mathcal{F}_k$. Let also $\mathcal{B} = \cup_{k=1}^{\infty} B^k$. Note that the set $\{B \times \mathcal{X}^{\mathbb{N}} : B \in B^{k,l}, k,l \in \mathbb{N}\}$ generates $\mathcal{F}$.

Definition 15 (distributional distance) *The distributional distance is defined for a pair of processes ρ_1, ρ_2 as follows*

$$D_{dd}(\rho_1,\rho_2) := \sum_{m,l=1}^{\infty} w_m w_l \sum_{B\in B^{m,l}} |\rho_1(B) - \rho_2(B)|, \tag{15}$$

where $w_k, k \in \mathbb{N}$ is a summable sequence of positive real weights (e.g., $w_j = 2^{-j}$).

Remark. A more general definition, which is not specific to time-series distributions, is to take any sequence $B_j \in \mathcal{F}_1$, $j \in \mathbb{N}$ of events that generate the sigma-algebra $\mathcal{F}$ of a probability space $(\mathcal{X}, \mathcal{F})$, and then define

$$D'_{dd}(\rho_1,\rho_2) := \sum_{j=1}^{\infty} w_j |\rho_1(B_j) - \rho_2(B_j)|; \tag{16}$$

see Gray (1988) for a general treatment. The latter definition is sometimes more convenient for theoretical analysis (Ryabko, 2012), while the distance (15), which makes explicit the marginal distributions on $\mathcal{X}^m$, $m \in \mathbb{N}$ and the level l of discretisation $B^{m,l}$ of each set $\mathcal{X}^m$, is more suited for time-series, and, specifically, for implementing algorithms, see Ryabko and Ryabko (2010), Khaleghi et al. (2012) and Khaleghi and Ryabko (2012).

In general, it is perhaps impossible to tell which distance, specifically, $D_{\mathbf{H}}$ or D_{dd}, should be preferred for which problem. Conceptually, one of the advantages of the telescope distance $D_{\mathbf{H}}$ is that one can use different sets $\mathbf{H}$—the choice that makes it adaptable to applications. Another is that one can reuse readily available classification methods for calculating its empirical estimates. One formal way to compare different metrics is to compare the resulting topologies. This is done in the end of this section.

8.2 sup Distances

A different way to construct a distance between time-series distributions based on their finite-dimensional marginals is to use the supremum instead of summation in (14):

$$d(\mu,\nu) = \sup_{k\in\mathbb{N}} d_k(\mu_k,\nu_k). \tag{17}$$

Some commonly used metrics are defined in the form (17) or have natural interpretations in this form, as the following two examples show.

Definition 16 (total variation) *For time-series distributions ν, μ the total variation distance between them is defined as $D_{tv}(\mu,\nu) := \sup_{A\in\mathcal{F}} |\mu(A) - \nu(A)|$.*

It is easy to see that $D_{tv}(\mu,\nu) = \sup_{k\in\mathbb{N}} \sup_{A\in\mathcal{F}_k} |\mu(A) - \nu(A)|$, so that the total variation distance has the form (17).

However, the total variation distance is not very useful for time-series distributions for the following two reasons. First of all, for stationary ergodic distributions it is degenerate: $D_{tv}(\mu,\nu) = 1$ if and only if $\mu \neq \nu$. This follows from the fact that any two different stationary ergodic distributions are singular. Such a distance could still be useful as a formalization of the problem of homogeneity testing. However, the problem of homogeneity testing is impossible to solve based on sampling for stationary ergodic distributions (and even for a smaller family of B processes, see below) (Ryabko, 2010b), so the use of this distance remains limited to more restrictive classes of distributions.

This hints at an intrinsic problem with distances defined in the form (17). The problem is in the difficulties to estimate such metrics based on sampling. At each time step t we observe only a sample of finite length, say n_t, and based on this we want to estimate a quantity that involves k-dimensional marginals for all k, including those with $k > n_t$. Considering a growing (with t) number of marginals for the estimate may be a route to take, but this turns out to be difficult to analyse, especially if no rates of convergence can be established for the set of time-series distributions at hand. This problem is highlighted by the example of the so-called $\bar{d}$ distance, whose definition follows.

Definition 17 ($\bar{d}$ distance) *Assume some distance δ over $\mathcal{X}$ is given. For two time-series distributions μ and ν define*

$$\bar{d}(\mu,\nu) := \sup_{k\in\mathbb{N}} \frac{1}{k} \inf_{p\in P} \sum_{i=1}^{k} \mathbf{E}_p \delta(x_i, y_i),$$

where P is the set of all distributions over $\mathcal{X}^k \times \mathcal{X}^k$ generating a pair of sequences $x_{1..k}, y_{1..k}$ whose marginal distributions are μ_k and ν_k correspondingly.

A process is called a *B-process* (or a Bernoulli process) if it is in the $\bar{d}$-closure of the set of all aperiodic stationary ergodic k-step Markov processes, where $k \in \mathbb{N}$. For more information on $\bar{d}$-distance and B-processes see Gray (1988) and Shields (1996). The set of B-processes is a strict subset of the set of all stationary ergodic time-series distributions. It turns out that $\bar{d}$ distance is impossible to estimate for the latter, while it can be estimated for the former (Ornstein and Weiss, 1990).

Theorem 18 (Ornstein and Weiss, 1990) *There exists an estimator $\hat{d}(X_{1..n}, Y_{1..n})$ such that, if $X_{1..n}, Y_{1..n}$ are generated by B-processes μ and ν then $\hat{d}(X_{1..n}, Y_{1..n}) \to \bar{d}(\mu,\nu)$ a.s. However, for any estimator $\hat{d}(X_{1..n}, Y_{1..n})$ there is a pair of stationary ergodic processes μ and ν such that $\limsup_{n\to\infty} |\hat{d}(X_{1..n}, Y_{1..n}) - \bar{d}(\mu,\nu)| > 1/2$.*

8.3 Comparison with the Distributional Distance

In this section we show that the telescope distance is stronger than the distributional distance in the topological sense. Since in fact both the telescope distance and the distributional distance are families of distances (the telescope distance depends on the sequence $\mathbf{H}$), we will fix a simple natural choice of each of these metrics. In general, different choices of parameters produce topologically non-equivalent metrics; it is easy to check that the analysis in this section extends to many other natural choices.

Thus, for the purpose of this section, let us fix $\mathcal{X} = \mathbb{R}$ and let H_k^0 be the set of halfspaces in $\mathcal{X}^k$. Denote $\mathbf{H}^0 := (\mathcal{H}_k^0 : k \in \mathbb{N})$. Clearly, these $\mathcal{H}_k$ satisfy all the conditions of the theorems of Sections 5 and 6.

For the distributional distance (Definition 15), set $B^{k,l}$ to be the partition of the set $\mathcal{X}^k$ into k-dimensional cubes with volume $h_l^k = (1/l)^k$. Denote D_{dd}^0 the distributional distance D_{dd} with this set of parameters.

Definition 19 *A metric d_1 is said to be* stronger *than a metric d_2 if any sequence that converges in d_1 also converges in d_2. If, in addition, d_2 is not stronger than d_1, then d_1 is called* strictly stronger.

Note that for the distributional distance, if we use the same sets B_k to generate the sigma algebras $\mathcal{X}^k$ then the distance defined by (15) is stronger than the distance defined by (16).

Theorem 20 *$D_{\mathbf{H}^0}$ is strictly stronger than D_{dd}^0.*

Proof Fix any $\varepsilon > 0$ and find a $T \in \mathbb{N}$ such that $\sum_{m,l>T} w_m w_l < \varepsilon$. Let ρ_i, $i \in \mathbb{N}$ be a sequence of process measures that converges in $D_{\mathbf{H}^0}$. Let A^k be the set of all complements to $\mathcal{X}^k$ of cubes with sides of length s, for all $s \in \mathbb{N}$. Note that any cube B in B_k, as well as any set A in A^k, can be obtained by intersecting $2k$ halfspaces. Therefore, we have

$$\sup_{B \in B^k \cup A^k} |\rho_i(B) - \rho_j(B)| \le 2k d_{\mathcal{H}_k}(\rho_i, \rho_j) \le 2k w_k^{-1} D_{\mathbf{H}^0}(\rho_i, \rho_j), \tag{18}$$

where the second inequality follows from the definition of $D_{\mathbf{H}^0}$. Observe that for each $i \in \mathbb{N}$ one can find a set $A_i \in A^k$ such that $\rho_i(A_i) < \varepsilon/2$. From this, (18) and the fact that the sequence ρ_i converges in $D_{\mathbf{H}^0}$, we conclude that there is a set $A \in A^k$ such that

$$\rho_i(A) < \varepsilon$$

for all $i \ge j_k$. For all $k, l \in \mathbb{N}$ one can find $M_{k,l} \in \mathbb{N}$ such that the complement of A (which is a cube in $\mathcal{X}^k$) is contained in the union of $M_{k,l}$ cubes from $B_{k,l}$. Let $M := \max_{k,l \le T} M_{k,l}$ and $J := \max_{i \le T} j_i$. Using (18) and the definition of the partitions $B^{k,l}$ we can derive

$$\sum_{B \in B^{k,l}, B \not\subseteq A_k} |\rho_i(B) - \rho_j(B)| \le 2MTw_T^{-1} D_{\mathbf{H}^0}(\rho_i, \rho_j)$$

for any $i, j \ge J$ and all $k, l \le T$. Increasing J if necessary to have $2MTw_T^{-1} D_{\mathbf{H}^0}(\rho_i, \rho_j) < \varepsilon$ for all $i, j \ge J$, we obtain

$$D_{dd}^0(\rho_i, \rho_j) \le \sum_{m,l=1}^{T} w_m w_l \sum_{B \in B^{m,l}, B \not\subseteq A_m} |\rho_i(B) - \rho_j(B)| + 2\varepsilon \le 3\varepsilon$$

for all $i, j > J$, which means that the sequence $\rho_i, i \in \mathbb{N}$ converges in D_{dd}^0. Thus, $D_{\mathbf{H}^0}$ is stronger than D_{dd}^0.

It remains to show that D_{dd}^0 is not stronger than $D_{\mathbf{H}^0}$. To see this, consider the following sequence of subsets of $\mathcal{X} = \mathbb{R}$. f is the dot $\{0\}$, and f_k is the interval $[0, 1/k]$, for each $k \in \mathbb{N}$. Define the distributions ν_j for $j \in \mathbb{N}$ as uniform on f_j, and let ν be concentrated on f; since we need time-series distributions, extend this i.i.d. for all $n \in \mathbb{N}$. It is easy to check that $\lim_{i \in \mathbb{N}} D_{dd}^0(\nu_i, \nu_0) = 0$ while $D_{\mathbf{H}^0}(\nu_i, \nu_0) = 1$ for all $i > 0$. ■

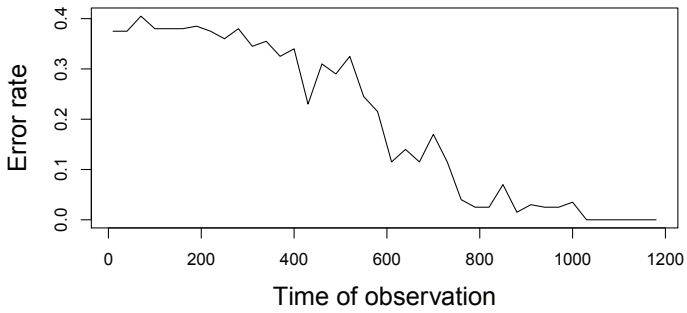


Figure 1: Error of two-class clustering using $\mathrm{TS}_{\mathrm{SVM}}$; 10 time series in each target cluster, averaged over 20 runs.

9. Experimental Evaluation

For experimental evaluation we chose the problem of time-series clustering. The average-linkage clustering is used, with the telescope distance between samples calculated using an SVM, as described in Section 4. In all experiments, SVM is used with radial basis kernel, with default parameters of libsvm (Chang and Lin, 2011). The parameters w_k in the definition of the telescope distance (Definition 2) are set to $w_k := k^{-2}$.

9.1 Synthetic Data

For the artificial setting we chose highly-dependent time-series distributions which have the same single-dimensional marginals and which cannot be well approximated by finite- or countable-state models. Variants of this family of distributions are standard examples in ergodic theory and dynamical systems (see, for example, Billingsley, 1965; Gray, 1988; Shields, 1996). The distributions $\rho(\alpha)$, $\alpha \in (0,1)$, are constructed as follows. Select $r_0 \in [0,1]$ uniformly at random; then, for each $i = 1..n$ obtain r_i by shifting r_{i-1} by α to the right, and removing the integer part. The time series $(X_1, X_2, \dots)$ is then obtained from r_i by drawing a point from a distribution law $\mathcal{N}_1$ if $r_i < 0.5$ and from $\mathcal{N}_2$ otherwise. $\mathcal{N}_1$ is a 3-dimensional Gaussian with mean of 0 and covariance matrix $\mathrm{Id} \times 1/4$. $\mathcal{N}_2$ is the same but with mean 1. If α is irrational[2] then the distribution $\rho(\alpha)$ is stationary ergodic, but does not belong to any simpler natural distribution family; in particular, it is not a B-processes (Shields, 1996). The single-dimensional marginal is the same for all values of α. The latter two properties make all parametric and most non-parametric methods inapplicable to this problem.

In our experiments, we use two process distributions $\rho(\alpha_i), i \in \{1,2\}$, with $\alpha_1 = 0.31...$, $\alpha_2 = 0.35...$,. The dependence of error rate on the length of time series is shown on Figure 1. One clustering experiment on sequences of length 1000 takes about 5 min. on a standard laptop.

9.2 Real Data

To demonstrate the applicability of the proposed methods to realistic scenarios, we chose the brain-computer interface data from BCI competition III (Millán, 2004). The data set consists of (preprocessed) BCI recordings of mental imagery: a person is thinking about one of three subjects

2. In the experiments we used a `longdouble` with a long mantissa

	s_1	s_2	s_3
TS_{SVM}	**84%**	**81%**	**61%**
DTW	46%	41%	36%
KCpA	79%	74%	61%
SVM	76%	69%	60%

Table 1: Clustering accuracy in the BCI data set. 3 subjects (columns), 4 methods (rows). Our method is TS_{SVM}.

(left foot, right foot, a random letter). Originally, each time series consisted of several consecutive sequences of different classes, and the problem was supervised: three time series for training and one for testing. We split each of the original time series into classes, and then used our clustering algorithm in a completely unsupervised setting. The original problem is 96-dimensional, but we used only the first 3 dimensions (using all 96 gives worse performance). The typical sequence length is 300. The performance is reported in Table 1, labelled TS_{SVM}. All the computation for this experiment takes approximately 6 minutes on a standard laptop.

The following methods were used for comparison. First, we used dynamic time wrapping (DTW) (Sakoe and Chiba, 1978) which is a popular base-line approach for time-series clustering. The other two methods in Table 1 are from the paper of Harchaoui et al. (2008). The comparison is not fully relevant, since the results of Harchaoui et al. (2008) are for different settings; the method KCpA was used in change-point estimation method (a different but also unsupervised setting), and SVM was used in a supervised setting. The latter is of particular interest since the classification method we used in the telescope distance is also SVM, but our setting is unsupervised (clustering). On this data set the telescope distance demonstrates better performance than the comparison methods, which indicates that it can be useful in real-world scenarios.

10. Outlook

We have proposed a binary-classifier-based metric and shown how it can be used to solve several problems concerning highly dependent time series. The consistency results obtained concern the use of the empirical risk minimizer as a binary classifier. For applications this suggests using classifiers that approximate empirical risk minimizers over target sets of (indicator) functions. It is easy to extend the definition of the metric so that any classifier can be used, including such classifiers as nearest-neighbours rules. However, in order to extend the obtained results to such classifiers, one would need to establish the consistency of the empirical estimates of the resulting metric between time-series distributions, which means extending the results concerning the corresponding classifiers from the i.i.d. samples to stationary ergodic time series. Note that, while consistency of the empirical estimates of the time-series metric used is sufficient for the analysis of the learning problems considered in this work, it is not sufficient for some other learning problems concerning dependent time series that rely on a metric between time-series distributions. For example, some change-point problems for stationary ergodic time series can be solved using the distributional distance (Ryabko and Ryabko, 2010; Khaleghi and Ryabko, 2012, 2013). It remains to see whether the same results can be obtained with the telescope distance and its generalizations.

Acknowledgments

This work is an extended version of the NIPS'12 paper (Ryabko and Mary, 2012). The authors are grateful to the anonymous reviewers for the numerous constructive comments that helped us to improve the paper. This research was partially supported by the French Ministry of Higher Education and Research, Nord-Pas-de-Calais Regional Council and FEDER through CPER 2007-2013, ANR project Lampada (ANR-09-EMER-007) and by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement 231495 (project CompLACS).

References

T. M. Adams and A. B. Nobel. Uniform approximation of Vapnik-Chervonenkis classes. *Bernoulli*, 18(4):1310–1319, 2012.

M.-F. Balcan, N. Bansal, A. Beygelzimer, D. Coppersmith, J. Langford, and G. Sorkin. Robust reductions from ranking to classification. In Nader Bshouty and Claudio Gentile, editors, *Learning Theory*, volume 4539 of *Lecture Notes in Computer Science*, pages 604–619. 2007.

M.-F. Balcan, A. Blum, and S. Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 671–680. ACM, 2008.

P. Billingsley. *Ergodic Theory and Information*. Wiley, New York, 1965.

D. Bosq. *Nonparametric Statistics for Stochastic Processes*. Estimation and Prediction. Springer, 1996.

Ch.-Ch. Chang and Ch.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

R. Fortet and E. Mourier. Convergence de la répartition empirique vers la répartition théoretique. *Ann. Sci. Ec. Norm. Super., III. Ser*, 70(3):267–285, 1953.

R. Gray. *Probability, Random Processes, and Ergodic Properties*. Springer Verlag, 1988.

M. Gutman. Asymptotically optimal classification for multiple tests with empirically observed statistics. *IEEE Transactions on Information Theory*, 35(2):402–408, 1989.

Z. Harchaoui, F. Bach, and E. Moulines. Kernel change-point analysis. In *Advances in Neural Information Processing Systems 21*, pages 609–616, 2008.

L. V. Kantorovich and G. S. Rubinstein. On a function space in certain extremal problems. *Dokl. Akad. Nauk USSR*, 115(6):1058–1061, 1957.

R.L. Karandikar and M. Vidyasagar. Rates of uniform convergence of empirical means with mixing processes. *Statistics and Probability Letters*, 58:297–307, 2002.

A. Khaleghi, D. Ryabko, J. Mary, and P. Preux. Online clustering of processes. In *AISTATS*, JMLR W&CP 22, pages 601–609, 2012.

A. Khaleghi and D. Ryabko. Locating changes in highly dependent data with unknown number of change points. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 3095–3103. 2012.

A. Khaleghi and D. Ryabko. Nonparametric multiple change point estimation in highly dependent time series. In *Proc. 24th International Conf. on Algorithmic Learning Theory (ALT'13)*, Singapre, 2013. Springer.

D. Kifer, Sh. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proc. the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB'04, pages 180–191, 2004.

A.N. Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *G. Inst. Ital. Attuari*, pages 83–91, 1933.

J. Langford, R. Oliveira, and B. Zadrozny. Predicting conditional quantiles via reduction to classification. In *Proc. of the 22th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2006.

J. del R. Millán. On the need for on-line learning in brain-computer interfaces. In *Proc. of the Int. Joint Conf. on Neural Networks*, 2004.

D.S. Ornstein and B. Weiss. How sampling reveals a process. *Annals of Probability*, 18(3):905–930, 1990.

D. Pollard. *Convergence of Stochastic Processes*. Springer, 1984.

B. Ryabko. Prediction of random sequences and universal coding. *Problems of Information Transmission*, 24:87–96, 1988.

B. Ryabko. Compression-based methods for nonparametric prediction and estimation of some characteristics of time series. *IEEE Transactions on Information Theory*, 55:4309–4315, 2009.

D. Ryabko. Clustering processes. In *Proc. the 27th International Conference on Machine Learning (ICML 2010)*, pages 919–926, Haifa, Israel, 2010a.

D. Ryabko. Discrimination between B-processes is impossible. *Journal of Theoretical Probability*, 23(2):565–575, 2010b.

D. Ryabko. On the relation between realizable and non-realizable cases of the sequence prediction problem. *Journal of Machine Learning Research*, 12:2161–2180, 2011.

D. Ryabko. Testing composite hypotheses about discrete ergodic processes. *Test*, 21(2):317–329, 2012.

D. Ryabko and B. Ryabko. Nonparametric statistical inference for ergodic processes. *IEEE Transactions on Information Theory*, 56(3):1430–1435, 2010.

D. Ryabko and J. Mary. Reducing statistical time-series problems to binary classification. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2069–2077. 2012.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

P. Shields. *The Ergodic Theory of Discrete Sample Paths*. AMS Bookstore, 1996.

R. J. Solomonoff. Complexity-based induction systems: comparisons and convergence theorems. *IEEE Trans. Information Theory*, IT-24:422–432, 1978.

V. M. Zolotarev. Probability metrics. *Theory of Probability and Its Applications.*, 28(2):264–287, 1983.

Journal of Machine Learning Research 14 (2013) 2857-2898 Submitted 1/13; Revised 7/13; Published 9/13

Perturbative Corrections for Approximate Inference in Gaussian Latent Variable Models

Manfred Opper OPPERM@CS.TU-BERLIN.DE
Department of Computer Science
Technische Universität Berlin
D-10587 Berlin, Germany

Ulrich Paquet ULRIPA@MICROSOFT.COM
Microsoft Research Cambridge
Cambridge CB1 2FB, United Kingdom

Ole Winther OWI@IMM.DTU.DK
Informatics and Mathematical Modelling
Technical University of Denmark
DK-2800 Lyngby, Denmark

Editor: Neil Lawrence

Abstract

Expectation Propagation (EP) provides a framework for approximate inference. When the model under consideration is over a latent Gaussian field, with the approximation being Gaussian, we show how these approximations can systematically be corrected. A perturbative expansion is made of the exact but intractable correction, and can be applied to the model's partition function and other moments of interest. The correction is expressed over the higher-order cumulants which are neglected by EP's local matching of moments. Through the expansion, we see that EP is correct to first order. By considering higher orders, corrections of increasing polynomial complexity can be applied to the approximation. The second order provides a correction in quadratic time, which we apply to an array of Gaussian process and Ising models. The corrections generalize to arbitrarily complex approximating families, which we illustrate on tree-structured Ising model approximations. Furthermore, they provide a polynomial-time assessment of the approximation error. We also provide both theoretical and practical insights on the exactness of the EP solution.

Keywords: expectation consistent inference, expectation propagation, perturbation correction, Wick expansions, Ising model, Gaussian process

1. Introduction

Expectation Propagation (EP) (Opper and Winther, 2000; Minka, 2001a,b) is part of a rich family of variational methods, which approximate the sums and integrals required for exact probabilistic inference by an optimization problem. Variational methods are perfectly amenable to probabilistic graphical models, as the nature of the optimization problem often allows it to be distributed across a graph. By relying on local computations on a graph, inference in very large probabilistic models becomes feasible.

Being an approximation, some error may invariably be introduced. This paper is specifically concerned with the error that arises when a Gaussian approximating family is used, and lays a systematic foundation for examining and correcting these errors. It follows on earlier work by the

authors (Opper et al., 2009). The error that arises when the free energy (the negative logarithm of the partition function or normalizer of the distribution) is approximated, may for instance be written as a Taylor expansion (Opper et al., 2009; Paquet et al., 2009). A pleasing property of EP is that, at its stationary point, the first order term of such an expansion is zero. Furthermore, the quality of the approximation can then be ascertained in polynomial time by including corrections beyond the first order, or beyond the standard EP solution. In general, the corrections improve the approximation when they are comparatively small, but can also leave a question mark on the quality of approximation when the lower-order terms are large.

The approach outlined here is by no means unique in correcting the approximation, as is evinced by cluster-based expansions (Paquet et al., 2009), marginal corrections for EP (Cseke and Heskes, 2011) and the Laplace approximation (Rue et al., 2009), and corrections to Loopy Belief Propagation (Chertkov and Chernyak, 2006; Sudderth et al., 2008; Welling et al., 2012).

1.1 Overview

EP is introduced in a general way in Section 3, making it clear how various degrees of complexity can be included in its approximating structure. The partition function will be used throughout the paper to explain the necessary machinery for correcting any moments of interest. In the experiments, corrections to the marginal and predictive means and variances are also shown, although the technical details for correcting moments beyond the partition function are relegated to Appendix D. The Ising model, which is cast as a Gaussian latent variable model in Section 2, will furthermore be used as a running example throughout the paper.

The key to obtaining a correction lies in isolating the "intractable quantity" from the "tractable part" (or EP solution) in the true problem. This is done by considering the cumulants of both: as EP locally matches lower-order cumulants like means and variances, the "intractable part" exists as an expression over the higher-order cumulants which are neglected by EP. This process is outlined in Section 4, which concludes with two useful results: a shift of the "intractable part" to be an average over complex Gaussian variables with *zero* diagonal relation matrix, and Wick's theorem, which allows us to evaluate the expectations of polynomials under centered Gaussian measures. As a last stage, the "intractable part" is expanded in Sections 5 and 7 to obtain corrections to various orders. In Section 6, we provide a theoretical analysis of the radius of convergence of these expansions.

Experimental evidence is presented in Section 8 on Gaussian process (GP) classification and (non-Gaussian) GP regression models. An insightful counterexample where EP diverges under increasing data, is also presented. Ising models are examined in Section 9.

Numerous additional examples, derivations, and material are provided in the appendices. Details on different EP approximations can be found in Appendix A, while corrections to tree-structured approximations are provided in Appendix B. In Appendix C we analytically show that the correction to a tractable example is zero. The main body of the paper deals with corrections to the partition function, while corrections to marginal moments are left to Appendix D. Finally, useful calculations of certain cumulants appear in Appendix E.

2. Gaussian Latent Variable Models

Let $\mathbf{x} = (x_1, \ldots, x_N)$ be an unobserved random variable with an intractable distribution $p(\mathbf{x})$. In the Gaussian latent variable model (GLVM) considered in this paper, terms $t_n(x_n)$ are combined over a

quadratic exponential $f_0(\mathbf{x})$ to give

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{n=1}^{N} t_n(x_n)\, f_0(\mathbf{x}) \tag{1}$$

with partition function (normalizer)

$$Z = \int \prod_{n=1}^{N} t_n(x_n)\, f_0(\mathbf{x})\, \mathrm{d}\mathbf{x}\ .$$

This model encapsulates many important methods used in statistical inference. As an example, f_0 can encode the covariance matrix of a Gaussian process (GP) prior on latent function observations x_n. In the case of GP classification with a class label $y_n \in \{-1,+1\}$ on a latent function evaluation x_n, the terms are typically probit link functions, for example

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{n=1}^{N} \Phi(y_n x_n)\, \mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{K})\ . \tag{2}$$

The probit function is the standard cumulative Gaussian density $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(z;0,1)\,\mathrm{d}z$. In this example, the partition function is not analytically tractable but for the one-dimensional case $N = 1$.

An Ising model can be constructed by letting the terms t_n restrict x_n to ± 1 (through Dirac delta functions). By introducing the symmetric coupling matrix $\mathbf{J}$ and field $\boldsymbol{\theta}$ into f_0, an Ising model can be written as

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{n=1}^{N}\left[\frac{1}{2}\delta(x_n+1)+\frac{1}{2}\delta(x_n-1)\right]\exp\left\{\frac{1}{2}\mathbf{x}^T\mathbf{J}\mathbf{x}+\boldsymbol{\theta}^T\mathbf{x}\right\}\ . \tag{3}$$

In the Ising model, the partition function Z is intractable, as it sums $f_0(\mathbf{x})$ over 2^N binary values of $\mathbf{x}$. In the variational approaches, the intractability is addressed by allowing approximations to Z and other marginal distributions, decreasing the computational complexity from being exponential to polynomial in N, which is typically cubic for EP.

3. Expectation Propagation

An approximation to Z can be made by allowing $p(\mathbf{x})$ in Equation (1) to factorize into a product of *factors* f_a. This factorization is not unique, and the structure of the factorization of $p(\mathbf{x})$ defines the complexity of the resulting approximation, resulting in different structures in the approximating distribution. Where GLVMs are concerned, a natural and computationally convenient choice is to use Gaussian factors g_a, and as such, the approximating distribution $q(\mathbf{x})$ in this paper will be Gaussian. Appendix A summarizes a number of factorizations for Gaussian approximations.

The tractability of the resulting inference method imposes a pragmatic constraint on the choice of factorization; in the extreme case $p(\mathbf{x})$ could be chosen as a single factor and inference would be exact. For the model in Equation (1), a three-term product may be factorized as $(t_1)(t_2)(t_3)$, which gives the typical GP setup. When a division is introduced and the term product factorizes as $(t_1t_2)(t_2t_3)/(t_2)$, the resulting free energy will be that of the tree-structured EC approximation (Opper and Winther, 2005). To therefore allow for regrouping, combining, splitting, and dividing terms, a power D_a is associated with each f_a, such that

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{a} f_a(\mathbf{x})^{D_a} \tag{4}$$

with intractable normalization (or partition function) $Z = \int \prod_a f_a(\mathbf{x})^{D_a} \mathrm{d}\mathbf{x}$.[1] Appendix A shows how the introduction of D_a lends itself to a clear definition of tree-structured and more complex approximations.

To define an approximation to p, terms g_a, which typically take an exponential family form, are chosen such that

$$q(\mathbf{x}) = \frac{1}{Z_q} \prod_a g_a(\mathbf{x})^{D_a} \tag{5}$$

has the same structure as p's factorization. Although not shown explicitly, f_a and g_a have a dependence on the *same* subset of variables $\mathbf{x}_a$. The optimal parameters of the g_a-term approximations are found through a set of auxiliary *tilted* distributions, defined by

$$q_a(\mathbf{x}) = \frac{1}{Z_a} \left(\frac{q(\mathbf{x}) f_a(\mathbf{x})}{g_a(\mathbf{x})} \right) . \tag{6}$$

Here a *single* approximating term g_a is replaced by an original term f_a. Assuming that this replacement leaves q_a still tractable, the parameters in g_a are determined by the condition that $q(\mathbf{x})$ and all $q_a(\mathbf{x})$ should be made as similar as possible. This is usually achieved by requiring that these distributions share a set of generalised moments which usually coincide with the sufficient statistics of the exponential family. For example with sufficient statistics $\phi(\mathbf{x})$ we require that

$$\langle \phi(\mathbf{x}) \rangle_{q_a} = \langle \phi(\mathbf{x}) \rangle_q \quad \text{for all } a . \tag{7}$$

Note that those factors f_a in $p(\mathbf{x})$ which are already in the exponential family, such as the Gaussian terms in examples above, can trivially be solved for by setting $g_a = f_a$. The partition function associated with this approximation is

$$Z_{\mathrm{EP}} = Z_q \prod_a Z_a^{D_a} . \tag{8}$$

Appendix A.2 shows that the moment-matching conditions must hold at a stationary point of $\log Z_{\mathrm{EP}}$. The EP algorithm iteratively updates the g_a-terms by enforcing q to share moments with each of the tilted distributions q_a; on reaching a fixed point all moments match according to Equation (7) (Minka, 2001a,b). Although Z_{EP} is defined in the terminology of EP, other algorithms may be required to solve for the fixed point, and Z_{EP}, as a free energy, can be derived from the saddle point of a set of self-consistent (moment-matching) equations (Opper and Winther, 2005; van Gerven et al., 2010; Seeger and Nickisch, 2010). We next make EP concrete by applying it to the Ising model, which will serve as a running example in the paper. The section is finally concluded with a discussion of the interpretation of EP.

3.1 EP for Ising Models

The Ising model in Equation (3) will be used as a running example throughout this paper. To make the technical developments more concrete, we will consider both the N-variate and bivariate cases. The bivariate case can be solved analytically, and thus allows for a direct comparison to be made between the exact and approximate solutions.

We use the factorized approximation as a running example, dividing $p(\mathbf{x})$ in Equation (3) into $N+1$ factors with $f_0(\mathbf{x}) = \exp\{\frac{1}{2}\mathbf{x}^T\mathbf{J}\mathbf{x} + \boldsymbol{\theta}^T\mathbf{x}\}$ and $f_n(x_n) = t_n(x_n) = \frac{1}{2}\delta(x_n+1) + \frac{1}{2}\delta(x_n-1)$, for

1. The factorization and EP energy function is expressed here in the form of Power EP (Minka, 2004).

$n = 1, \ldots, N$ (see Appendix A for generalizations). We consider the Gaussian exponential family such that $g_n(x_n) = \exp\{\lambda_{n1} x_n - \frac{1}{2}\lambda_{n2} x_n^2\}$ and $g_0(\mathbf{x}) = f_0(\mathbf{x})$. The approximating distribution from Equation (5), $q(\mathbf{x}) \propto f_0(\mathbf{x}) \prod_{n=1}^{N} g_n(x_n)$, is thus a *full* multivariate Gaussian density, which we write as $q(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$.

3.1.1 MOMENT MATCHING

The moment matching condition in Equation (7) involves only the mean and variance if $q(\mathbf{x})$ fully factorizes according to $p(\mathbf{x})$'s terms. We therefore only need to match the mean and variances of marginals of $q(\mathbf{x})$ and the tilted distribution $q_n(\mathbf{x})$ in Equation (6). The tilted distribution may be decomposed into a Gaussian and a discrete part as $q_n(\mathbf{x}) = q_n(\mathbf{x}_{\setminus n}|x_n) q_n(x_n)$, where the vector $\mathbf{x}_{\setminus n}$ consists of all variables apart from x_n. We may marginalize out $\mathbf{x}_{\setminus n}$ and write $q_n(x_n)$ in terms of two factors:

$$q_n(x_n) \propto \underbrace{\frac{1}{2}\Big[\delta(x_n+1) + \delta(x_n-1)\Big]}_{f_n(\mathbf{x}) = t_n(x_n)} \underbrace{\exp\Big\{\gamma x_n - \tfrac{1}{2}\Lambda x_n^2\Big\}}_{\propto \int d\mathbf{x}_{\setminus n}\, q(\mathbf{x})/g_n(\mathbf{x})}, \tag{9}$$

where we dropped the dependency of γ and Λ on n for notational simplicity. Through some manipulation, the tilted distribution is equivalent to

$$q_n(x_n) = \frac{1+m_n}{2}\,\delta(x_n - 1) + \frac{1-m_n}{2}\,\delta(x_n+1)\,, \quad m_n = \tanh(\gamma) = \frac{e^{\gamma} - e^{-\gamma}}{e^{\gamma} + e^{-\gamma}}\,. \tag{10}$$

This discrete distribution has mean m_n and variance $1 - m_n^2$. By adapting the parameters of $g_n(x_n)$ using for example the EP algorithm, we aim to match the mean and variance of the marginal $q(x_n)$ (of $q(\mathbf{x})$) to the mean and variance of $q_n(x_n)$. The reader is referred to Section 9 for benchmarked results for the Ising model.

3.1.2 ANALYTIC BIVARIATE CASE

Here we shall compare the exact result with EP and the correction for the simplest non-trivial model, the $N = 2$ Ising model with no external field

$$p(\mathbf{x}) = \frac{1}{4}\Big(\delta(x_1 - 1) + \delta(x_1 + 1)\Big)\Big(\delta(x_2 - 1) + \delta(x_2 + 1)\Big)\, e^{J x_1 x_2}\,.$$

In order to solve the moment matching conditions we observe that the mean values must be zero because the distribution is symmetric around zero. Likewise the linear term in the approximating factors disappears and we can write $g_n(x_n) = \exp\{-\lambda x_n^2/2\}$ and $q(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{0}, \boldsymbol{\Sigma})$ with $\boldsymbol{\Sigma} = \begin{bmatrix} \lambda & -J \\ -J & \lambda \end{bmatrix}^{-1}$. The moment matching condition for the variances, $1 = \Sigma_{nn}$, turns into a second order equation with solution $\lambda = \frac{1}{2}\Big[J^2 + \sqrt{J^4 + 4}\Big]$. We can now insert this solution into the expression for the EP partition function in Equation (8). By expanding the result to the second order in J^2, we find that

$$\log Z_{\mathrm{EP}} = -\frac{1}{2} + \frac{1}{2}\sqrt{1 + 4J^2} - \frac{1}{2}\log\left(\frac{1}{2}(1 + \sqrt{1 + 4J^2})\right) = \frac{J^2}{2} - \frac{J^4}{4} + \ldots\,.$$

Comparing with the exact expression

$$\log Z = \log\cosh(J) = \frac{J^2}{2} - \frac{J^4}{12} + \ldots$$

we see that EP gives the correct J^2 coefficient, but the J^4 coefficient comes out wrong. In Section 4 we investigate how cumulant corrections can correct for this discrepancy.

3.2 Two Explanations Why Gaussian EP is Often Very Accurate

EP, as introduced above, is an algorithm. The justification for the algorithm put forward by Minka and adopted by others (see for example recent textbooks by Bishop 2006, Barber 2012 and Murphy 2012) is useful for explaining the steps in the algorithm but may be misleading in order to explain why EP often provides excellent accuracy in estimation of marginal moments and Z.

The general justification for EP (Minka, 2001a,b) is based upon a minimization of Kullback-Leiber (KL) divergences. Ideally, one would determine the approximating distribution $q(\mathbf{x})$ as the minimizer of $\mathrm{KL}(p\|q)$ in an exponential family of (in our case, Gaussian) densities. Since this is not possible—it would require the computation of exact moments—we instead iteratively minimize "local" KL-divergences $\mathrm{KL}(q_a\|q)$, between the tilted distribution q_a and q, with respect to g_a (appearing in q). This leads to the moment matching conditions in Equation (7). The argument for this procedure is essentially that this will ensure that the approximation q will capture high density regions of the intractable posterior p. Obviously, this argument cannot be applied to Ising models because the exact and approximate distributions are very different, with the former being discrete due to the Dirac δ-functions that constrain $x_n = \pm 1$ to be binary variables. Even though the optimization still implies moment matching, this discrete-continuous discrepancy makes local KL-divergences $\mathrm{KL}(q_a\|q)$ *infinite*!

In order to justify the usefulness of EP for Ising models we therefore need an alternative argument. Our argument is entirely restricted to *Gaussian* EP for our extended definition of GLVMs and do not extend to approximations with other exponential families. In the following, we will discuss these assumptions in inference approximations that preceded the formulation of EP, in order to provide a possibly more relevant justification of the method. Although this justification is not strictly necessary for practically using EP nor corrections to EP, it nevertheless provides a good starting point for understanding both.

The argument goes back to the mathematical analysis of the Sherrington-Kirkpatrick (SK) model for a disordered magnet (a so-called spin glass) (Sherrington and Kirckpatrick, 1975). For this Ising model, the couplings $\mathbf{J}$ are drawn at random from a Gaussian distribution. An important contribution in the context of inference for this model (the computations of partition functions and average magnetizations) was the work of Thouless et al. (1977) who derived *self-consistency equations* which are assumed to be valid with a probability (with respect to the drawing of random couplings) approaching one as the number of variables x_n grows to infinity. These so-called Thouless-Anderson-Palmer (TAP) equations are closely related to the EP moment matching conditions of Equation (7), but they differ by partly relying on the specific assumption of the randomness of the couplings. Self-consistency equations equivalent to the EP moment matching conditions which avoided such assumptions on the statistics of the random couplings were first derived by Opper and Winther (2000) by using a so-called cavity argument (Mézard et al., 1987). A new important contribution of Minka (2001a) was to provide an efficient algorithmic recipe for solving these equations.

We will now sketch the main idea of the cavity argument for the GLVM. Let $\mathbf{x}_{\setminus n}$ ("$\mathbf{x}$ without n") denote the complement to x_n, that is $\mathbf{x} = \mathbf{x}_{\setminus n} \cup x_n$. Without loss of generality we will take the quadratic exponential term to be written as $f_0(\mathbf{x}) \propto \exp(-\mathbf{x}^T\mathbf{J}\mathbf{x}/2)$. With similar definitions of $\mathbf{J}_{\setminus n}$,

the exact marginal distribution of x_n may be written as

$$\begin{aligned} p_n(x_n) &= \frac{1}{Z} t_n(x_n) \int \exp\left\{-\frac{1}{2}\mathbf{x}^T \mathbf{J}\mathbf{x}\right\} \prod_{n' \neq n} t_{n'}(x_{n'}) \, \mathrm{d}\mathbf{x}_{\setminus n} \\ &= \frac{t_n(x_n)}{Z} \, \mathrm{e}^{-J_{nn} x_n^2/2} \int \exp\left\{-x_n \sum_{n' \neq n} J_{nn'} x_{n'} - \frac{1}{2}\mathbf{x}_{\setminus n}^T \mathbf{J}_{\setminus n} \mathbf{x}_{\setminus n}\right\} \prod_{n' \neq n} t_{n'}(x_{n'}) \, \mathrm{d}\mathbf{x}_{\setminus n} \, . \end{aligned}$$

It is clear that $p_n(x_n)$ depends entirely on the statistics of the random variable $h_n \equiv \sum_{n' \neq n} J_{nn'} x_{n'}$. This is the total *'field'* created by all other 'magnetic moments' $x_{n'}$ in the 'cavity' opened once x_n has been removed from the system. In the context of densely connected models with weak couplings, we can appeal to the central limit theorem[2] to approximate h_n by a Gaussian random variable with mean γ_n and variance V_n. When looking at the influence of the remaining variables $\mathbf{x}_{\setminus n}$ on x_n, the non-Gaussian details of their distribution have been washed out in the marginalization. Integrating out the Gaussian random variable h_n gives the Gaussian cavity field approximation to the marginal distribution:

$$\begin{aligned} p_n(x_n) &\approx \mathrm{const} \cdot t_n(x_n) \, \mathrm{e}^{-J_{nn} x_n^2/2} \int \mathrm{e}^{-x_n h} \, \mathcal{N}(h; \gamma_n, V_n) \, \mathrm{d}h \\ &= \mathrm{const} \cdot t_n(x_n) \exp\left\{-x_n \gamma_n - \frac{1}{2}(J_{nn} - V_n) x_n^2\right\} \, . \end{aligned}$$

This is precisely of the form of the marginal tilted distribution $q_n(x_n)$ of Equation (9) as given by Gaussian EP. In the cavity formulation, $q(\mathbf{x})$ is simply a *placeholder* for the sufficient statistics of the individual Gaussian cavity fields. So we may observe cases, with the Ising model or bounded support factors being the prime examples, where EP gives essentially correct results for the marginal distributions of the x_n and of the partition function Z, while $q(\mathbf{x})$ gives a poor or even meaningless (in the sense of KL divergences) approximation to the multivariate posterior. Note however, that the entire *covariance matrix* of the x_n can be computed simply from a derivative of the free energy (Opper and Winther, 2005) resulting in an approximation of this covariance by that of $q(\mathbf{x})$. This may indicate that a good EP approximation of the free energy may also result in a good approximation to the full covariance. The near exactness of EP (as compared to exhaustive summation) in Section 9 therefore shows the central limit theorem at work. Conversely, mediocre accuracy or even failure of Gaussian EP, as also observed in our simulations in Sections 8.3 and 9, may be attributed to breakdown of the Gaussian cavity field assumption. Exact inference on the strongest couplings as considered for the Ising model in Section 9 is one way to alleviate the shortcoming of the Gaussian cavity field assumption.

4. Corrections to EP

The Z_{EP} approximation can be corrected in a principled approach, which traces the following outline:

1. The exact partition function Z is re-written in terms of Z_{EP}, scaled by a correction factor $R = Z/Z_{\mathrm{EP}}$. This correction factor R encapsulates the intractability in the model, and contains a "local marginal" contribution by each f_a (see Section 4.1).

2. In the context of sparsely connected models, other cavity arguments lead to loopy belief propagation.

2. A "handle" on R is obtained by writing it in terms of the cumulants (to be defined in Section 4.2) of $q(\mathbf{x})$ and $q_a(\mathbf{x})$ from Equations (5) and (6). As $q_a(\mathbf{x})$ and $q(\mathbf{x})$ share their two first cumulants, the mean and covariance from the moment matching condition in Equation (7), a cumulant expansion of R will be in terms of *higher-order* cumulants (see Section 4.2).

3. R, defined in terms of cumulant differences, is written as a complex Gaussian average. Each factor f_a contributes a complex random variable $\mathbf{k}_a$ in this average (see Section 4.3).

4. Finally, the cumulant differences are used as "small quantities" in a Taylor series expansion of R, and the leading terms are kept (see Sections 5 and 7).

   The series expansion is in terms of a complex expectation with a *zero* "self-relation" matrix, and this has two important consequences. Firstly, it causes all first order terms in the Taylor expansion to disappear, showing that $Z_{\rm EP}$ is correct to first order. Secondly, due to Wick's theorem (introduced in Section 4.4), these zeros will contract the expansion by making many other terms vanish.

The strategy that is presented here can be re-used to correct other quantities of interest, like marginal distributions or the predictive density of new data when $p(\mathbf{x})$ is a Bayesian probabilistic model. These corrections are outlined in Appendix D.

4.1 Exact Expression for Correction

We define the (intractable) correction R as $Z = RZ_{\rm EP}$. We can derive a useful expression for R in a few steps as follows: First we solve for f_a in Equation (6), and substitute this into Equation (4) to obtain

$$\prod_a f_a(\mathbf{x})^{D_a} = \prod_a \left(\frac{Z_a q_a(\mathbf{x}) g_a(\mathbf{x})}{q(\mathbf{x})} \right)^{D_a} = Z_{\rm EP}\, q(\mathbf{x}) \prod_a \left(\frac{q_a(\mathbf{x})}{q(\mathbf{x})} \right)^{D_a} . \tag{11}$$

We introduce $F(\mathbf{x})$

$$F(\mathbf{x}) \equiv \prod_a \left(\frac{q_a(\mathbf{x})}{q(\mathbf{x})} \right)^{D_a}$$

to derive the expression for the correction $R = Z/Z_{\rm EP}$ by integrating Equation (11):

$$R = \int q(\mathbf{x}) F(\mathbf{x})\, \mathrm{d}\mathbf{x} \,, \tag{12}$$

where we have used $Z = \int \prod_a f_a(\mathbf{x})^{D_a}\, \mathrm{d}\mathbf{x}$. Similarly we can write:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_a f_a(\mathbf{x})^{D_a} = \frac{Z_{\rm EP}}{Z}\, q(\mathbf{x}) F(\mathbf{x}) = \frac{1}{R}\, q(\mathbf{x}) F(\mathbf{x}) \,. \tag{13}$$

Corrections to the marginal and predictive densities of $p(\mathbf{x})$ can be computed from this formulation. This expression will become especially useful because the terms in $F(\mathbf{x})$ turn out to be "local", that is, they only depend on the marginals of the variables associated with factor a. Let $f_a(\mathbf{x})$ depend on the subset $\mathbf{x}_a$ of $\mathbf{x}$, and let $\mathbf{x}_{\setminus a}$ ("$\mathbf{x}$ without a") denote the remaining variables. The distributions in Equations (5) and (6) differ only with respect to their marginals on $\mathbf{x}_a$, $q_a(\mathbf{x}_a)$ and $q(\mathbf{x}_a)$, and therefore

$$\frac{q_a(\mathbf{x})}{q(\mathbf{x})} = \frac{q(\mathbf{x}_{\setminus a}|\mathbf{x}_a) q_a(\mathbf{x}_a)}{q(\mathbf{x}_{\setminus a}|\mathbf{x}_a) q(\mathbf{x}_a)} = \frac{q_a(\mathbf{x}_a)}{q(\mathbf{x}_a)} \,.$$

Now we can rewrite $F(\mathbf{x})$ in terms of marginals:

$$F(\mathbf{x}) = \prod_a \left(\frac{q_a(\mathbf{x}_a)}{q(\mathbf{x}_a)} \right)^{D_a} . \tag{14}$$

The key quantity, then, is F, after which the key operation is to compute its expected value. The rest of this section is devoted to the task of obtaining a "handle" on F.

4.2 Characteristic Functions and Cumulants

The distributions present in each of the ratios in $F(\mathbf{x})$ in Equation (14) share their first two cumulants, mean and covariance. Cumulants and cumulant differences are formally defined in the next paragraph. This simple observation has a crucial consequence: As the $q(\mathbf{x}_a)$'s are Gaussian and do not contain any higher order cumulants (three and above), F can be expressed in terms of the higher cumulants of the *marginals* $q_a(\mathbf{x}_a)$. When the term-product approximation is fully factorized, these are simply cumulants of *one*-dimensional distributions.

Let N_a be the number of variables in subvector $\mathbf{x}_a$. In the examples presented in this work, N_a is one or two. Furthermore, let $\mathbf{k}_a$ be an N_a-dimensional vector $\mathbf{k}_a = (k_1, \ldots, k_{N_a})_a$. The characteristic function of q_a is

$$\chi_a(\mathbf{k}_a) = \int \mathrm{e}^{i\mathbf{k}_a^T \mathbf{x}_a} q_a(\mathbf{x}_a)\, \mathrm{d}\mathbf{x}_a = \left\langle \mathrm{e}^{i\mathbf{k}_a^T \mathbf{x}_a} \right\rangle_{q_a} , \tag{15}$$

and is obtained through the Fourier transform of the density. Inversely,

$$q_a(\mathbf{x}_a) = \frac{1}{(2\pi)^{N_a}} \int \mathrm{e}^{-i\mathbf{k}_a^T \mathbf{x}_a} \chi_a(\mathbf{k}_a)\, \mathrm{d}\mathbf{k}_a . \tag{16}$$

The cumulants $c_{\boldsymbol{\alpha} a}$ of q_a are the coefficients that appear in the Taylor expansion of $\log \chi_a(\mathbf{k}_a)$ around the zero vector,

$$c_{\boldsymbol{\alpha} a} = \left[(-i)^l \left(\frac{\partial}{\partial \mathbf{k}_a} \right)^{\boldsymbol{\alpha}} \log \chi_a(\mathbf{k}_a) \right]_{\mathbf{k}_a = \mathbf{0}} .$$

By this definition of $c_{\boldsymbol{\alpha} a}$, the Taylor expansion of $\log \chi_a(\mathbf{k}_a)$ is

$$\log \chi_a(\mathbf{k}_a) = \sum_{l=1}^{\infty} i^l \sum_{|\boldsymbol{\alpha}|=l} \frac{c_{\boldsymbol{\alpha} a}}{\boldsymbol{\alpha}!} \mathbf{k}_a^{\boldsymbol{\alpha}} .$$

Some notation was introduced in the above two equations to facilitate manipulating a multivariate series. The vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{N_a})$, with $\alpha_j \in \mathbb{N}_0$, denotes a multi-index on the elements of $\mathbf{k}_a$. Other notational conventions that employ $\boldsymbol{\alpha}$ (writing k_j instead of k_{aj}) are:

$$|\boldsymbol{\alpha}| = \sum_j \alpha_j , \qquad \mathbf{k}_a^{\boldsymbol{\alpha}} = \prod_j k_j^{\alpha_j} , \qquad \boldsymbol{\alpha}! = \prod_j \alpha_j ! , \qquad \left(\frac{\partial}{\partial \mathbf{k}_a} \right)^{\boldsymbol{\alpha}} = \prod_j \frac{\partial^{\alpha_j}}{\partial k_j^{\alpha_j}} .$$

For example, when $N_a = 2$, say for the edge-factors in a spanning tree, the set of multi-indices $\boldsymbol{\alpha}$ where $|\boldsymbol{\alpha}| = 3$ are $(3,0)$, $(2,1)$, $(1,2)$, and $(0,3)$.

There are two characteristic functions that come into play in $F(\mathbf{x})$ and R in Equation (13). The first is that of the tilted distribution, $\log \chi_a(\mathbf{k}_a)$, and the other is the characteristic function of the

EP marginal $q(\mathbf{x}_a)$, defined as $\chi(\mathbf{k}_a) = \langle e^{i\mathbf{k}_a^T \mathbf{x}_a} \rangle_q$. By virtue of matching the first two moments, and $q(\mathbf{x}_a)$ being Gaussian with cumulants $c'_{\alpha a}$,

$$r_a(\mathbf{k}_a) = \log \chi_a(\mathbf{k}_a) - \log \chi(\mathbf{k}_a) = \sum_{l \geq 1} i^l \sum_{|\alpha|=l} \frac{c_{\alpha a} - c'_{\alpha a}}{\alpha!} \mathbf{k}_a^\alpha$$
$$= \sum_{l \geq 3} i^l \sum_{|\alpha|=l} \frac{c_{\alpha a}}{\alpha!} \mathbf{k}_a^\alpha \tag{17}$$

contains the remaining higher-order cumulants where the tilted and approximate distributions *differ*. All our subsequent derivations rest upon moment matching being attained. This especially means that one cannot use the derived corrections if EP has not converged.

4.2.1 ISING MODEL EXAMPLE

The cumulant expansion for the discrete distribution in Equation (10) becomes

$$\log \chi_n(k_n) = \log \int dx_n \, e^{ik_n x_n} q_n(x_n) = \log \left(\frac{1+m}{2} e^{ik_n} + \frac{1-m}{2} e^{-ik_n} \right)$$
$$= imk_n - \frac{1}{2!}(1-m^2)k_n^2 - \frac{i}{3!}(-2m+2m^3)k_n^3 + \frac{1}{4!}(-2+8m^2-6m^4)k_n^4 + \cdots$$

(we're compactly writing m for m_n), from which the cumulants are obtained as

$$c_{1n} = m \,, \qquad c_{4n} = -2 + 8m^2 - 6m^4 \,,$$
$$c_{2n} = 1 - m^2 \,, \qquad c_{5n} = 16m - 40m^3 + 24m^5 \,,$$
$$c_{3n} = -2m + 2m^3 \,, \qquad c_{6n} = 16 - 136m^2 + 240m^4 - 120m^6 \,.$$

4.3 The Correction as a Complex Expectation

The expected value of F, which is required for the correction, has a dependence on a product of ratios of distributions $q_a(\mathbf{x}_a)/q(\mathbf{x}_a)$. In the preceding section it was shown that the contributing distributions share lower-order statistics, allowing a twofold simplification. Firstly, the ratio q_a/q will be written as a single quantity that depends on r_a, which was introduced above in Equation (17). Secondly, we will show that it is natural to shift integration variables into the complex plane, and rely on complex Gaussian random variables (meaning that both real and imaginary parts are jointly Gaussian). These complex random variables that define the r_a's have a peculiar property: they have a zero self-relation matrix! This property has important consequences in the resulting expansion.

4.3.1 COMPLEX EXPECTATIONS

Assume that $q(\mathbf{x}_a) = \mathcal{N}(\mathbf{x}_a; \boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a)$ and $q_a(\mathbf{x}_a)$ share the same mean and covariance, and substitute $\log \chi_a(\mathbf{k}_a) = r_a(\mathbf{k}_a) + \log \chi(\mathbf{k}_a)$ in the definition of q_a in Equation (16) to give

$$\frac{q_a(\mathbf{x}_a)}{q(\mathbf{x}_a)} = \frac{\int e^{-i\mathbf{k}_a^T \mathbf{x}_a + r_a(\mathbf{k}_a)} \chi(\mathbf{k}_a) \, d\mathbf{k}_a}{\int e^{-i\mathbf{k}_a^T \mathbf{x}_a} \chi(\mathbf{k}_a) \, d\mathbf{k}_a} \,. \tag{18}$$

Although the $\mathbf{k}_a$ variables have not been introduced as random variables, we find it natural to *interpret them as such*, because the rules of expectations over Gaussian random variables will be

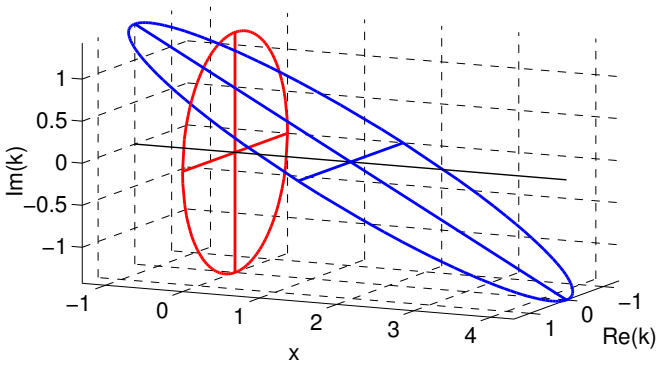


Figure 1: Equation (20) shifts $\mathbf{k}_a$ to the complex plane. In the simplest case the joint density $p(k|x)\,q(x)$ is $x \sim \mathcal{N}(\mu, \sigma^2)$, $\Re(k) \sim \mathcal{N}(0, \sigma^{-2})$ and equality $\Im(k) = -\sigma^{-2}(x-\mu)$. Notice that $\Re(k)$'s variance is the *inverse* of that of x. The joint density is a two-dimensional flat ellipsoidal pancake that lives in three dimensions: x and the complex k plane (tilted ellipsoid). Integrating over x gives the marginal over a complex k, which is *still* a two-dimensional random variable (upright ellipsoid). The marginal has $\Im(k) \sim \mathcal{N}(0, \sigma^{-2})$, and hence k has relation $\left\langle (\Re(k) + i\Im(k))^2 \right\rangle = \sigma^{-2} - \sigma^{-2} = 0$ and variance $\left\langle k\bar{k} \right\rangle = 2\sigma^{-2}$.

extremely helpful in developing the subsequent expansions. We will therefore write $q_a(\mathbf{x}_a)/q(\mathbf{x}_a)$ as an expectation of $\exp r_a(\mathbf{k}_a)$ over a density $p(\mathbf{k}_a|\mathbf{x}_a) \propto \mathrm{e}^{-i\mathbf{k}_a^T \mathbf{x}_a} \chi(\mathbf{k}_a)$:

$$\frac{q_a(\mathbf{x}_a)}{q(\mathbf{x}_a)} = \Big\langle \exp r_a(\mathbf{k}_a) \Big\rangle_{\mathbf{k}_a|\mathbf{x}_a} . \tag{19}$$

By substituting $\log \chi(\mathbf{k}_a) = i\boldsymbol{\mu}_a^T \mathbf{k}_a - \mathbf{k}_a^T \boldsymbol{\Sigma}_a \mathbf{k}_a/2$ into Equation (18), we see that $p(\mathbf{k}_a|\mathbf{x}_a)$ can be viewed as Gaussian, but not for real random variables! We have to consider $\mathbf{k}_a$ as Gaussian random variables with a real and an imaginary part with

$$\Re(\mathbf{k}_a) \sim \mathcal{N}\Big(\Re(\mathbf{k}_a)\,;\, \mathbf{0},\, \boldsymbol{\Sigma}_a^{-1}\Big) , \qquad \Im(\mathbf{k}_a) = -\boldsymbol{\Sigma}_a^{-1}(\mathbf{x}_a - \boldsymbol{\mu}_a) .$$

For the purpose of computing the expectation in Equation (19), $\mathbf{k}_a|\mathbf{x}_a$ is a degenerate complex Gaussian that shifts the coefficients $\mathbf{k}_a$ into the complex plane. The expectation of $\exp r_a(\mathbf{k}_a)$ is therefore taken over Gaussian random variables that have $q(\mathbf{x}_a)$'s *inverse* covariance matrix as their (real) covariance! As shorthand, we write

$$p(\mathbf{k}_a|\mathbf{x}_a) = \mathcal{N}\Big(\mathbf{k}_a\,;\, -i\boldsymbol{\Sigma}_a^{-1}(\mathbf{x}_a - \boldsymbol{\mu}_a)\,,\, \boldsymbol{\Sigma}_a^{-1}\Big) . \tag{20}$$

Figure 1 illustrates a simple density $p(\mathbf{k}_a|\mathbf{x}_a)$, showing that the imaginary component is a deterministic function of $\mathbf{x}_a$. Once $\mathbf{x}_a$ is averaged out of the joint density $p(\mathbf{k}_a|\mathbf{x}_a)\,q(\mathbf{x}_a)$, a *circularly symmetric complex Gaussian* distribution over $\mathbf{k}_a$ remains. It is circularly symmetric as $\langle \mathbf{k}_a \rangle = \mathbf{0}$, relation matrix $\left\langle \mathbf{k}_a \mathbf{k}_a^T \right\rangle = \mathbf{0}$, and covariance matrix $\left\langle \mathbf{k}_a \overline{\mathbf{k}_a}^T \right\rangle = 2\boldsymbol{\Sigma}_a^{-1}$ (notation $\bar{k}$ indicates the complex conjugate of k). For the purpose of computing the expected values with Wick's theorem (following in Section 4.4 below), we *only* need the relations $\left\langle \mathbf{k}_a \mathbf{k}_b^T \right\rangle$ for pairs of factors a and b. All of these will be derived next:

According to Equation (12), a further expectation over $q(\mathbf{x})$ is needed, after integrating over $\mathbf{k}_a$, to determine R. These variables will be *combined* into complex random variables to make the averages in the expectation easier to derive. By substituting Equation (19) into Equation (12), R is equal to

$$R = \left\langle F(\mathbf{x}) \right\rangle_{\mathbf{x}\sim q(\mathbf{x})} = \left\langle \prod_a \left\langle \exp r_a(\mathbf{k}_a) \right\rangle_{\mathbf{k}_a|\mathbf{x}_a}^{D_a} \right\rangle_{\mathbf{x}} . \tag{21}$$

When $\mathbf{x}$ is given, the $\mathbf{k}_a$-variables are independent. However, when they are averaged over $q(\mathbf{x})$, the $\mathbf{k}_a$-variables become coupled. They are zero-mean complex Gaussians

$$\langle \mathbf{k}_a \rangle = \left\langle \langle \mathbf{k}_a \rangle_{\mathbf{k}_a|\mathbf{x}_a} \right\rangle_{\mathbf{x}} = \left\langle -i\boldsymbol{\Sigma}_a^{-1}(\mathbf{x}_a - \boldsymbol{\mu}_a) \right\rangle_{\mathbf{x}} = \mathbf{0}$$

and are coupled with a zero self-relation matrix! In other words, if $\boldsymbol{\Sigma}_{ab} = \mathrm{cov}(\mathbf{x}_a, \mathbf{x}_b)$, the expected values $\left\langle \mathbf{k}_a \mathbf{k}_b^T \right\rangle$ between the variables in the set $\{\mathbf{k}_a\}$ are

$$\begin{aligned} \left\langle \mathbf{k}_a \mathbf{k}_b^T \right\rangle &= \left\langle \left\langle \mathbf{k}_a \mathbf{k}_b^T \right\rangle_{\mathbf{k}_{a,b}|\mathbf{x}} \right\rangle_{\mathbf{x}} + i^2 \boldsymbol{\Sigma}_a^{-1} \left\langle (\mathbf{x}_a - \boldsymbol{\mu}_a)(\mathbf{x}_b - \boldsymbol{\mu}_b)^T \right\rangle_{\mathbf{x}} \boldsymbol{\Sigma}_b^{-1} \\ &= \begin{cases} \mathbf{0} & \text{if } a = b \\ -\boldsymbol{\Sigma}_a^{-1}\boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_b^{-1} & \text{if } a \neq b \end{cases} . \end{aligned} \tag{22}$$

Complex Gaussian random variables are additionally characterized by $\left\langle \mathbf{k}_a \overline{\mathbf{k}_b}^T \right\rangle$. However, these expectations are not required for computing and simplifying the expansion of $\log R$ in Section 5, and are not needed for the remainder of this paper. Figure 2 illustrates the structure of the resulting relation matrix $\left\langle \mathbf{k}_a \mathbf{k}_b^T \right\rangle$ for two different factorizations of the same distribution. Each factor f_a contributes a $\mathbf{k}_a$ variable, such that the tree-structured approximation's relation matrix will be larger than that of the fully factorized one.

Section 5 shows that when $D_a = 1$, the above expectation can be written directly over $\{\mathbf{k}_a\}$ and expanded. In the general case, discussed in Section 7, the inner expectation is first expanded (to treat the D_a powers) before computing an expectation over $\{\mathbf{k}_a\}$. In both cases the expectation will involve polynomials in k-variables. The expected values of Gaussian polynomials can be evaluated with Wick's theorem.

4.4 Wick's Theorem

Wick's theorem provides a useful formula for mixed central moments of Gaussian variables. Let $k_{n_1}, \ldots, k_{n_\ell}$ be real or complex centered jointly Gaussian variables, noting that they do not have to be different. Then

$$\langle k_{n_1} \cdots k_{n_\ell} \rangle = \sum \prod_{\eta} \left\langle k_{i_\eta} k_{j_\eta} \right\rangle , \tag{23}$$

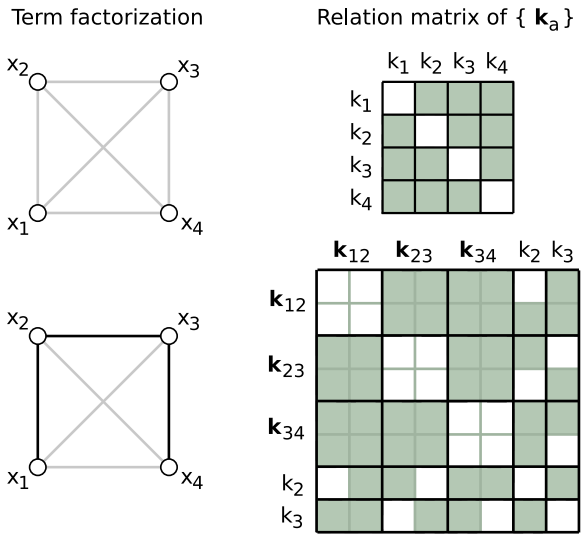


Figure 2: The relation matrices between $\mathbf{k}_a$ for two factorizations of $\prod_{n=1}^{4} t_n(x_n)$: the top illustration is for $t_1t_2t_3t_4$, while the bottom illustration is of a tree structure $(t_1t_2)(t_2t_3)(t_3t_4)/t_2/t_3$. The white squares indicate a zero relation matrix $\left\langle \mathbf{k}_a \mathbf{k}_b^T \right\rangle$, with the *diagonal* being zero. From the properties of Equation (22) there are additional zeros in the tree structure's relation matrix, where edge and node factors share variables. The factor $f_0 = g_0$ is shadowed in grey in the left-hand figures, and can make $q(\mathbf{x})$ densely connected.

where the sum is over all partitions of $\{n_1, \ldots, n_\ell\}$ into disjoint pairs $\{i_\eta, j_\eta\}$. If $\ell = 2m$ is even, then there are $(2m)!/(2^m m!) = (2m-1)!!$ such partitions.[3] If ℓ is odd, then there are none, and the expectation in Equation (23) is zero.

Consider the one-dimensional variable $k \sim \mathcal{N}(k; 0, \sigma^2)$. Wick's theorem states that $\langle k^\ell \rangle = (\ell - 1)!!\,\sigma^\ell$ if ℓ is even, and $\langle k^\ell \rangle = 0$ if ℓ is odd. In other words, $\langle k^3 \rangle = 0$, $\langle k^4 \rangle = 3(\sigma^2)^2$, $\langle k^6 \rangle = 15(\sigma^2)^3$, and so forth.

5. Factorized Approximations

In the fully factorized approximation, with $f_n(x_n) = t_n(x_n)$, the exact distribution in Equation (13) depends on the *single node marginals* $F(\mathbf{x}) = \prod_n q_n(x_n)/q(x_n)$. Following Equation (21), the correction to the free energy

$$R = \left\langle \prod_n \left\langle \exp r_n(k_n) \right\rangle_{k_n|x_n} \right\rangle_{\mathbf{x}} = \left\langle \exp \left[\sum_n r_n(k_n) \right] \right\rangle_{\mathbf{k}} \tag{24}$$

is taken directly over the centered complex-valued Gaussian random variables $\mathbf{k} = (k_1, \ldots, k_N)$, which have a relations

$$\langle k_m k_n \rangle = \begin{cases} 0 & \text{if } m = n \\ -\Sigma_{mn}/(\Sigma_{mm}\Sigma_{nn}) & \text{if } m \neq n \end{cases} . \tag{25}$$

3. The double factorial is $(2m-1)!! = (2m-1) \times (2m-3) \times (2m-5) \times \cdots 1$.

In the section to follow, all expectations shall be with respect to $\mathbf{k}$, which will be dropped where it is clear from the context.

Thus far, R is re-expressed in terms of site contributions. The expression in Equation (24) is exact, albeit still intractable, and will be treated through a power series expansion. Other quantities of interest, like marginal distributions or moments, can similarly be expressed exactly, and then expanded (see Appendix D).

5.1 Second Order Correction to $\log R$

Assuming that the r_n's are small on average with respect to $\mathbf{k}$, Equation (24) is expanded and the lower order terms kept:

$$\begin{aligned}\log R = \log\left\langle \exp\left[\sum_n r_n(k_n)\right]\right\rangle &= \sum_n \langle r_n\rangle + \frac{1}{2}\left\langle\left(\sum_n r_n\right)^2\right\rangle - \frac{1}{2}\left(\sum_n \langle r_n\rangle\right)^2 + \cdots \\ &= \frac{1}{2}\sum_{m\neq n}\langle r_m r_n\rangle + \cdots \end{aligned} \tag{26}$$

The simplification in the second line is a result of the variance terms being zero from Equation (25). The single marginal terms also vanish (and hence EP is correct to first order) because both $\langle k_n\rangle = 0$ and $\langle k_n^2\rangle = 0$.

This result can give us a hint in which situations the corrections are expected to be small:

- Firstly, the r_n could be small for values of k_n where the density of $\mathbf{k}$ is not small. For example, under a zero noise Gaussian process classification model, $q_n(x_n)$ equals a step function $t_n(x_n)$ times a Gaussian, where the latter often has small variance compared to the mean. Hence, $q_n(x_n)$ should be very close to a Gaussian.

- Secondly, for systems with weakly (posterior) dependent variables x_n we might expect that the log partition function $\log Z$ would scale approximately linearly with N, the number of variables. Since terms with $m = n$ vanish in the computation of $\ln R$, there are no corrections that are *proportional to* N when Σ_{mn} is sufficiently small as $N \to \infty$. Hence, the dominant contributions to $\log Z$ should already be included in the EP approximation. However, Section 8.3 illustrates an example where this need not be the case.

The expectation $\langle r_m r_n\rangle$, as it appears in Equation (26), is treated by substituting r_n with its cumulant expansion $r_n(k_n) = \sum_{l\geq 3} i^l c_{ln} k_n^l / l!$ from Equation (17). Wick's theorem now plays a pivotal role in evaluating the expectations that appear in the expansion:

$$\begin{aligned}\langle r_m(k_m) r_n(k_n)\rangle &= \sum_{l,s\geq 3} i^{l+s}\frac{c_{ln}c_{sm}}{l!s!}\langle k_m^s k_n^l\rangle \\ &= \sum_{l\geq 3} i^{2l} l!\frac{c_{ln}c_{sm}}{(l!)^2}\langle k_m k_n\rangle^l \\ &= \sum_{l\geq 3}\frac{c_{lm}c_{ln}}{l!}\left(\frac{\Sigma_{mn}}{\Sigma_{mm}\Sigma_{nn}}\right)^l .\end{aligned} \tag{27}$$

The second line above follows from contractions in Wick's theorem. All the *self-pairing terms*, when for example one of the l k_n's is paired with another k_n in Equation (23), are zero because

$\langle k_n^2 \rangle = 0$. To therefore get a non-zero result for $\langle k_m^s k_n^l \rangle$, using Equation (23), *each* factor k_n has to be paired with some factor k_m, and this is possible only when $l = s$. Wick's theorem sums over all pairings, and there are $l!$ ways of pairing a k_n with a k_m, giving the result in Equation (27). Finally, plugging Equation (27) into Equation (26) gives the second order correction

$$\log R = \frac{1}{2} \sum_{m \neq n} \sum_{l \geq 3} \frac{c_{lm} c_{ln}}{l!} \left(\frac{\Sigma_{mn}}{\Sigma_{mm} \Sigma_{nn}} \right)^l + \cdots . \tag{28}$$

5.1.1 ISING EXAMPLE CONTINUED

We can now compute the second order $\log R$ correction for the $N = 2$ Ising model example of Section 3.1. The covariance matrix has $\Sigma_{nn} = 1$ from moment matching and $\Sigma_{12} = J/(\lambda^2 - J^2)$ with $\lambda = \frac{1}{2}\left[J^2 + \sqrt{J^4 + 4}\right]$. The uneven terms in the cumulant expansion derived in Section 4.2.1 disappear because $m = 0$. The first nontrivial term is therefore $l = 4$ which gives a contribution of $\frac{1}{2} \times 2 \times \frac{c_4^2}{4!} \Sigma_{12}^4 = \frac{(-2)^2}{4!} \Sigma_{12}^4 = \frac{1}{6} \Sigma_{12}^4$. In Section 3.1, we saw that $\log Z - \log Z_{\mathrm{EP}} = \frac{J^4}{6}$ plus terms of order J^6 and higher. To lowest order in J we have $\Sigma_{12} = J$ and thus $\log R = \frac{J^4}{6}$ which exactly cancels the lowest order error of EP.

5.2 Corrections to Other Quantities

The schema given here is applicable to any other quantity of interest, be it marginal or predictive distributions, or the marginal moments of $p(\mathbf{x})$. The cumulant corrections for the marginal moments are derived in Appendix D; for example, the correction to the marginal mean μ_i of an approximation $q(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is

$$\langle x_i \rangle_{p(\mathbf{x})} - \mu_i = \sum_{l \geq 3} \sum_{j \neq n} \frac{\Sigma_{ij}}{\Sigma_{jj}} \frac{c_{l+1,j} c_{ln}}{l!} \left(\frac{\Sigma_{jn}}{\Sigma_{jj} \Sigma_{nn}} \right)^l + \cdots , \tag{29}$$

while the correction to the marginal covariance is

$$\begin{aligned} \langle (x_i - \mu_i)(x_{i'} - \mu_{i'}) \rangle_{p(\mathbf{x})} - \Sigma_{ii'} = & \sum_{l \geq 3} \sum_{j \neq n} \frac{\Sigma_{ij} \Sigma_{i'j}}{\Sigma_{jj}^2} \frac{c_{l+2,j} c_{ln}}{l!} \left(\frac{\Sigma_{jn}}{\Sigma_{jj} \Sigma_{nn}} \right)^l \\ & + \sum_{l \geq 3} \sum_{j \neq n} \frac{\Sigma_{ij}}{\Sigma_{jj}} \frac{\Sigma_{i'n}}{\Sigma_{nn}} \frac{c_{lj} c_{ln}}{l!} \left(\frac{\Sigma_{jn}}{\Sigma_{jj} \Sigma_{nn}} \right)^{l-1} + \cdots . \end{aligned} \tag{30}$$

5.3 Edgeworth-Type Expansions

To simplify the expansion of Equation (24), we integrated (combined) degenerate complex Gaussians $k_n | x_n$ over $q(\mathbf{x})$ to obtain fully complex Gaussian random variables $\{k_n\}$. We've then relied on $\langle k_n^2 \rangle = 0$ to simplify the expansion of $\log R$.

The expectations $\langle k_n^2 \rangle = 0$ are closely related to the orthogonality of Hermite polynomials, and this can be employed in an alternative derivation. In particular, one can *first* make a Taylor expansion of $\exp r_n(k_n)$ around zero, giving complex-valued polynomials in $\{k_n\}$. When the inner average in Equation (24) is then taken over $k_n | x_n$, a real-valued series of Hermite polynomials in $\{x_n\}$ arises. These polynomials are orthogonal under $q(\mathbf{x})$. The series that describes the tilted distribution $q_n(x_n)$ is equal to the product of $q(x_n)$ and an expansion of polynomials for the higher-cumulant *deviation*

from a Gaussian density. This line of derivation gives an Edgeworth expansion for*each* factor's tilted distribution.

As a second step, Equation (24) couples the product of separate Edgeworth expansions (one for each factor) together by requiring an outer average over $q(\mathbf{x})$. The orthogonality of Hermite polynomials under $q(\mathbf{x})$ now come into play: it allows products of orthogonal polynomials under $q(\mathbf{x})$ to integrate to zero. This is similar to contractions in Wick's theorem, where $\langle k_n^2 \rangle = 0$ allows us to simplify Equation (27). Although it is not the focus of this work, an example of such a derivation appears in Appendix C.1.

6. Radius of Convergence

We may hope that in practice the low order terms in the cumulant expansions will account already for the dominant contributions. But will such an expansion actually converge when extended to arbitrary orders? While we will leave a more general answer to future research, we can at least give a partial result for the example of the Ising model. Let $\mathbf{D} = \mathrm{diag}(\mathbf{\Sigma})$, the diagonal of the covariance matrix of the EP approximation $q(\mathbf{x})$. We prove here that a cumulant expansion for R will converge when the eigenvalues of $\mathbf{D}^{-1/2}\mathbf{\Sigma}\mathbf{D}^{-1/2}$—which has diagonal values of one—are bounded between zero and two.

In practice we've found that even if the largest of these eigenvalues grows with N, the second-order correction gives a remarkable improvement. This, with the results in Figure 6, lead us to believe that the power series expansion is often divergent. It may well be that our expansions are only of an asymptotic type (Boyd, 1999) for which the summation of only a certain number of terms might give an improvement whereas further terms would lead to worse results. It leads to a paradoxical situation, which seems common when interesting functions are computed: On the one hand we may have a series which does not converge, but in many ways is more practical; on the other hand one might obtain an expansion that converges, but only impractically. Quoting George F. Carrier's rule from Boyd (1999):

> *Divergent series converge faster than convergent series because they don't have to converge.*

For this, we do not yet have a clear-cut answer.

6.1 A Formal Expression for the Cumulant Expansion to All Orders

To discuss the question when our expansion will converge when extended to arbitrary orders, we introduce a single extra parameter λ into R, which controls the strength of the contribution of cumulants. Expanded into a series in powers of λ, contributions of cumulants of *total* order l are multiplied by a factor λ^l, for example $\lambda^l c_{nl}$ or $\lambda^{k+l} c_{nk} c_{nl}$. Of course, at the end of the calculation, we set $\lambda = 1$. This approach is obviously achieved by replacing

$$r_n(k_n) \to r_n(\lambda k_n)$$

in Equation (24). Hence, we define

$$R(\lambda) = \left\langle \exp\left[\sum_n r_n(\lambda k_n)\right] \right\rangle_{\mathbf{k}} = \left\langle \exp\left[\sum_n r_n(k_n)\right] \right\rangle_{\mathbf{k}'}$$

where

$$\langle k'_m k'_n \rangle = \begin{cases} 0 & \text{if } m = n \\ -\lambda^2 \Sigma_{mn}/(\Sigma_{mm}\Sigma_{nn}) & \text{if } m \neq n \end{cases} .$$

By working backwards, and expressing everything by the original densities over x_n, the correction can be written as

$$R(\lambda) = \left\langle \prod_n \frac{q_n(x_n)}{q(x_n)} \right\rangle_{q_\lambda(\mathbf{x})} , \tag{31}$$

where the density $q_\lambda(\mathbf{x})$ is a multivariate Gaussian with mean $\boldsymbol{\mu}$ and covariance given by

$$\boldsymbol{\Sigma}_\lambda = \mathbf{D} + z(\boldsymbol{\Sigma} - \mathbf{D}) ,$$

where $\mathbf{D} = \text{diag}(\boldsymbol{\Sigma})$ and $z = \lambda^2$. Hence, we see that the expansion in powers of λ is actually equivalent to an expansion in products of nondiagonal elements of $\boldsymbol{\Sigma}$.

Noticing that as $R(\lambda)$ depends on λ through the density $q_\lambda(\mathbf{x}) \propto |\boldsymbol{\Sigma}_\lambda|^{-1/2} \mathrm{e}^{-\frac{1}{2}\mathbf{x}^\top \boldsymbol{\Sigma}_\lambda^{-1}\mathbf{x}}$, we can see by expressing $\boldsymbol{\Sigma}_\lambda^{-1}$ in terms of eigenvalues and eigenvectors that for any *fixed* $\mathbf{x}$, $q_\lambda(\mathbf{x})$ is an analytic function of the *complex variable* z as long as $\boldsymbol{\Sigma}_\lambda$ is positive definite. Since

$$\boldsymbol{\Sigma}_\lambda = \mathbf{D}^{1/2} \left\{ \mathbf{I} + z \left(\mathbf{D}^{-1/2} \boldsymbol{\Sigma} \mathbf{D}^{-1/2} - \mathbf{I} \right) \right\} \mathbf{D}^{1/2}$$

this is equivalent to the condition that the matrix $\mathbf{I} + z(\mathbf{D}^{-1/2}\boldsymbol{\Sigma}\mathbf{D}^{-1/2} - \mathbf{I})$ is positive definite. Introducing γ_i, the eigenvalues of $\mathbf{D}^{-1/2}\boldsymbol{\Sigma}\mathbf{D}^{-1/2}$, positive definiteness fails when for the first time $1 + z(\gamma_i - 1) = 0$. Thus the series for $q_\lambda(\mathbf{x})$ is convergent for

$$|z| < \min_i \frac{1}{|1 - \gamma_i|} .$$

Setting $z = 1$, this is equivalent to the condition

$$1 < \min_i \frac{1}{|1 - \gamma_i|} .$$

This means that the eigenvalues have to fulfil $0 < \gamma_i < 2$. Unfortunately, we can not conclude from this condition that pointwise convergence of $q_\lambda(\mathbf{x})$ for each $\mathbf{x}$ leads to convergence of $R(\lambda)$ (which is an integral of $q_\lambda(\mathbf{x})$ over all $\mathbf{x}$!). However, in cases where the integral eventually becomes a finite sum, such as the Ising model, pointwise convergence in $\mathbf{x}$ leads to convergence of $R(\lambda)$.

6.1.1 ISING MODEL EXAMPLE

From Section 4.2.1 the tilted distribution for the running example Ising model is $q_n(x_n) = \frac{1}{2}[\delta(x_n + 1) + \delta(x_n - 1)]$, and hence $q(x_n) = \frac{1}{(2\pi)^{1/2}} \mathrm{e}^{-x_n^2/2}$. As each $q(x_n)$ is a unit-variance Gaussian, $\mathbf{D} = \text{diag}(\boldsymbol{\Sigma}) = \mathbf{I}$. Hence $\mathbf{D}^{-1/2}\boldsymbol{\Sigma}\mathbf{D}^{-1/2} = \boldsymbol{\Sigma}$ and

$$R(\lambda) = \frac{1}{\sqrt{|(1-\lambda^2)\mathbf{I} + \lambda^2 \boldsymbol{\Sigma}|}} \frac{\mathrm{e}^{N/2}}{2^N} \sum_{\mathbf{x} \in \{-1,1\}^N} \exp\left[-\frac{1}{2} \mathbf{x}^T \left((1-\lambda^2)\mathbf{I} + \lambda^2 \boldsymbol{\Sigma} \right)^{-1} \mathbf{x} \right]$$

follows from Equation (31). The arguments of the previous section show that the *radius of convergence* of $R(\lambda)$ is determined by the condition that the matrix $\mathbf{I} + \lambda^2(\boldsymbol{\Sigma} - \mathbf{I})$ is positive definite or the eigenvalues l_i of $\boldsymbol{\Sigma}$ fulfil $|l_i - 1| \leq 1/\lambda^2$.

In the $N = 2$ case, $\mathbf{\Sigma} = \begin{pmatrix} 1 & c \\ c & 1 \end{pmatrix}$ with $c = c(J) \in]-1,1[$ which has eigenvalues $1-c$ and $1+c$, meaning that cumulant expansion for $R(\lambda)$ is convergent for the $N = 2$ Ising model. For $N > 2$, it is easy to show that this is not necessarily true. Consider the 'isotropic' Ising model with $J_{ij} = J$ and zero external field, then $\Sigma_{ii} = 1$ and $\Sigma_{ij} = c$ for $i \neq j$ with $c = c(J) \in]-1/(N-1),1[$. The eigenvalues are now $1+(N-1)c$ and $1-c$ (the latter with degeneracy $N-1$). For finite c, the largest eigenvalue will scale with N and thus be larger than the upper value of two that would be required for convergence. Scaling with N for the largest eigenvalue of $\mathbf{D}^{-1/2}\mathbf{\Sigma}\mathbf{D}^{-1/2}$ is also observed in the Ising model simulations Section 9.

We conjecture that convergence of the cumulant series for $R(\lambda)$ also implies convergence of the series for $\log R(\lambda)$ but leave an investigation of this point to future research. We only illustrate this point for the $N = 2$ Ising model case, where we have the explicit formula

$$\log R(\lambda) = 1 - \frac{1}{2}\log\left(1-\lambda^4 c^2\right) - \frac{1}{1-\lambda^4 c^2} + \log\cosh\left(\frac{\lambda^2 c}{1-\lambda^4 c^2}\right) .$$

As can be easily seen, an expansion in λ converges for $c^2\lambda^4 < 1$ which gives the same radius of convergence $|c| < 1$ as for the expansion of R.

7. General Approximations

The general approximations differ from the factorized approximation in that an expansion in terms of expectations under $\{\mathbf{k}_a\}$ doesn't immediately arise. Consider R in Equation (21): Its inner expectations are over $\mathbf{k}_a|\mathbf{x}$, and outer expectations are over $\mathbf{x}$. First take the binomial expansion of the *inner* expectation, and keep it to second order in r_a:

$$\begin{aligned}
\left\langle \mathrm{e}^{r_a(\mathbf{k}_a)} \right\rangle^{D_a}_{\mathbf{k}_a|\mathbf{x}} &= \left(1 + \langle r_a\rangle + \frac{1}{2}\left\langle r_a^2\right\rangle + \cdots\right)^{D_a} \\
&= 1 + D_a\left[\langle r_a\rangle + \frac{1}{2}\left\langle r_a^2\right\rangle + \cdots\right] + \frac{D_a(D_a-1)}{2}\left[\langle r_a\rangle + \frac{1}{2}\left\langle r_a^2\right\rangle + \cdots\right]^2 + \cdots \\
&= 1 + D_a\langle r_a\rangle + \frac{D_a}{2}\left\langle r_a^2\right\rangle + \frac{D_a(D_a-1)}{2}\langle r_a\rangle^2 + \cdots .
\end{aligned}$$

Notice that $r_a(\mathbf{k}_a)$ can be complex, but $\langle r_a(\mathbf{k}_a)\rangle_{\mathbf{k}_a|\mathbf{x}}$, as it appears in the above expansion, is real-valued. Using this result, again expand $\langle \prod_a \langle \mathrm{e}^{r_a}\rangle^{D_a}_{\mathbf{k}_a|\mathbf{x}}\rangle_{\mathbf{x}}$. The correction to $\log R$, up to second order, is

$$\begin{aligned}
\log R = \frac{1}{2}\sum_{a\neq b} D_a D_b \left\langle \langle r_a(\mathbf{k}_a)\rangle_{\mathbf{k}_a|\mathbf{x}} \langle r_b(\mathbf{k}_b)\rangle_{\mathbf{k}_b|\mathbf{x}} \right\rangle_{\mathbf{x}} \\
+ \frac{1}{2}\sum_a D_a(D_a-1)\left\langle \langle r_a(\mathbf{k}_a)\rangle^2_{\mathbf{k}_a|\mathbf{x}} \right\rangle_{\mathbf{x}} + \cdots .
\end{aligned} \tag{32}$$

In the above relation the first-order terms all disappeared as $\langle\langle r_a(\mathbf{k}_a)\rangle\rangle = 0$. Terms involving $\langle\langle r_a(\mathbf{k}_a)^2\rangle\rangle = 0$ similarly disappear, as every polynomial in the expansion $r_a(\mathbf{k}_a)^2$ averages to zero. This is a general case of Equation (26), in which $D_n = 1$ for all factors. In Appendix B we show how to use the general result for the case where the factorization is a tree and our factors are edges (pairs) and nodes (single variables).

8. Gaussian Process Results

One of the most important applications of EP is to statistical models with Gaussian process (GP) priors, where $\mathbf{x}$ is a latent variable with Gaussian prior distribution with a kernel matrix $\mathbf{K}$ as covariance $\mathbb{E}[\mathbf{x}\mathbf{x}^T] = \mathbf{K}$.

It is well known that for many models, like GP classification, inference with EP is on par with MCMC ground truth (Kuss and Rasmussen, 2005). Section 8.1 underlines this case, and shows corrections to the partition function on the USPS data set over a range of kernel hyperparameter settings.

A common inference task is to predict the output for previously unseen data. Under a GP regression model, a key quantity is the predictive mean function. The predictive mean is analytically tractable when the latent function is corrupted with Gaussian noise to produce observations y_n. This need not be the case; in Section 8.2 we examine the problem of quantized regression, where the noise model is non-Gaussian with sharp discontinuities. We show practically how the corrections transfer to other moments, like the predictive mean. Through it, we arrive at a hypothetical rule of thumb: if the data isn't "sensible" under the (probabilistic) model of interest, there is no guarantee for EP giving satisfactory inference.

Armed with the rule of thumb, Section 8.3 constructs an insightful counterexample where the EP estimate diverges or is far from ground truth with more data. Divergence in the partition function is manifested in the initial correction terms, giving a test for the approximation accuracy that doesn't rely on any Monte Carlo ground truth.

8.1 Gaussian Process Classification

The GP classification model arises when we observe N data points $\mathbf{s}_n$ with class labels $y_n \in \{-1, 1\}$, and model y through a latent function x with a GP prior. The likelihood terms for y_n are assumed to be $t_n(x_n) = \Phi(y_n x_n)$, where $\Phi(\cdot)$ denotes the cumulative Normal density.

An extensive MCMC evaluation of EP for GP classification on various data sets was given by Kuss and Rasmussen (2005), showing that the log marginal likelihood of the data can be approximated remarkably well. As shown by Opper et al. (2009), an even more accurate estimation of the approximation error is given by considering the second order correction in Equation (28). For GPC we generally found that the $l = 3$ term dominates $l = 4$, and we do not include any higher cumulants here.

Figure 3 illustrates the correction to $\log R$, with $l = 3, 4$, on the binary subproblem of the USPS 3's vs. 5's digits data set, with $N = 767$. This is the same set-up of Kuss and Rasmussen (2005) and Opper et al. (2009), using the kernel $k(\mathbf{s}, \mathbf{s}') = \sigma^2 \exp(-\frac{1}{2}\|\mathbf{s} - \mathbf{s}'\|^2/\ell^2)$, and we refer the reader to both papers for additional and complimentary figures and results. We evaluated Equation (28) on a similar grid of $\log \ell$ and $\log \sigma$ values. For the same grid values we obtained Monte Carlo estimates of $\log Z$, and hence $\log R$. The correction, compared to the magnitude of the $\log Z$ grids by Kuss and Rasmussen (2005), is remarkably small, and underlines their findings on the accuracy of EP for GPC.

The correction from Equation (28), as computed here, is $O(N^2)$, and compares favorably to $O(N^3)$ complexity of EP for GPC.

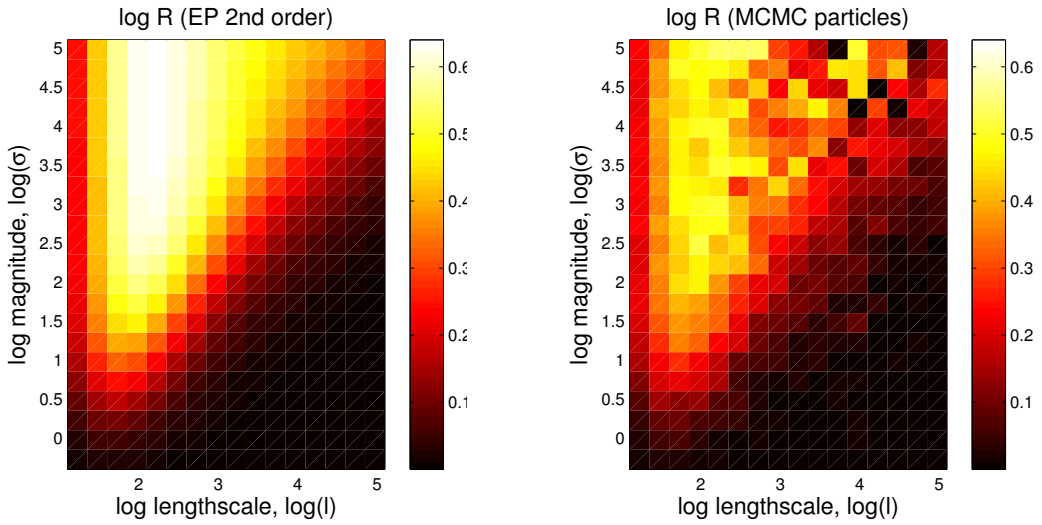


Figure 3: A comparison of $\log R$ using a perturbation expansion of Equation (28) against Monte Carlo estimates of $\log R$, using the USPS data set from Kuss and Rasmussen (2005). The second order correction to $\log R$, with $l = 3,4$, is used on the *left*; the *right* plot uses a Monte Carlo estimate of $\log R$.

8.2 Uniform Noise Regression

We turn our attention to a regression problem, that of learning a latent function $x(\mathbf{s})$ from inputs $\{\mathbf{s}_n\}$ and matching real-valued observations $\{y_n\}$. A frequent nonparametric treatment assumes that $x(\mathbf{s})$ is *a priori* drawn from a GP prior with covariance function $k(\mathbf{s},\mathbf{s}')$, from which a corrupted version y is observed. Analytically tractable inference is no longer possible in this model when the observation noise is non-Gaussian. Some scenarios include that of quantized regression, where y_n is formed by rounding $x(\mathbf{s}_n)$ to, say, the nearest integer, or where $x(\mathbf{s})$ indicates a robot's path in a control problem, with conditions to stay within certain "wall" bounds. In these scenarios the latent function $x(\mathbf{s}_n)$ can be reconstructed from y_n by adding sharply discontinuous uniformly random $\mathcal{U}[-a,a]$ noise,

$$p(\mathbf{x}) = \frac{1}{Z}\prod_n \mathbb{I}\Big[|x_n - y_n| < a\Big]\,\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{K})\ .$$

We now assume an EP approximation $q(\mathbf{x}) = \mathcal{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$, which can be obtained by using the moment calculations in Appendix E.2. To simplify the exposition of the predictive marginal, we follow the notation of Rasmussen and Williams (2005, Chapter 3) and let $\boldsymbol{\lambda}_n = (\tau_n,\nu_n)$, so that the final EP approximation multiplies g_n terms $\prod_n \exp\{-\frac{1}{2}\tau_n x_n^2 + \nu_n x_n\}$ into a joint Gaussian $\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{K})$.

8.2.1 MAKING PREDICTIONS FOR NEW DATA

The latent function $x(\mathbf{s}_*)$ at any new input $\mathbf{s}_*$ is obtained by the predictive marginal $q(x_*)$ of $q(\mathbf{x},x_*)$. The marginal $q(x_*)$—given below in Equation (34)—is directly obtained from the EP approximation

$q(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. However, the correction to its mean, as was given in Equation (29), requires covariances Σ_{*n}, which are derived here.

Let $\kappa_* = k(\mathbf{s}_*, \mathbf{s}_*)$, and $\mathbf{k}_*$ be a vector containing the covariance function evaluations $k(\mathbf{s}_*, \mathbf{s}_n)$. Again following Rasmussen and Williams (2005)'s notation, let $\tilde{\boldsymbol{\Sigma}}$ be the diagonal matrix containing $1/\tau_n$ along its diagonal. The EP covariance, on the inclusion of x_*, is

$$\begin{aligned}\boldsymbol{\Sigma}_* &= \left(\begin{bmatrix} \mathbf{K} & \mathbf{k}_* \\ \mathbf{k}_*^T & \kappa_* \end{bmatrix}^{-1} + \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}^{-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} \right)^{-1} \\ &= \begin{bmatrix} \boldsymbol{\Sigma} & \mathbf{k}_* - \mathbf{K}(\mathbf{K}+\tilde{\boldsymbol{\Sigma}})^{-1}\mathbf{k}_* \\ \mathbf{k}_*^T - \mathbf{k}_*^T(\mathbf{K}+\tilde{\boldsymbol{\Sigma}})^{-1}\mathbf{K} & \kappa_* - \mathbf{k}_*^T(\mathbf{K}+\tilde{\boldsymbol{\Sigma}})^{-1}\mathbf{k}_* \end{bmatrix}, \end{aligned} \tag{33}$$

with $\boldsymbol{\Sigma} = \mathbf{K} - \mathbf{K}(\mathbf{K}+\tilde{\boldsymbol{\Sigma}})^{-1}\mathbf{K}$. There is no observation associated with $\mathbf{s}_*$, hence $\tau_* = 0$ in the first line above, and its inclusion has $c_{l*} = 0$ for $l \geq 3$. The second line follows by computing matrix partitioned inverses twice on $\boldsymbol{\Sigma}_*$. The joint EP approximation for any new input point $\mathbf{s}_*$ is directly obtained as

$$q(\mathbf{x}, x_*) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x} \\ x_* \end{bmatrix}; \begin{bmatrix} \boldsymbol{\mu} \\ \mathbf{k}_*^T\mathbf{K}^{-1}\boldsymbol{\mu} \end{bmatrix}, \boldsymbol{\Sigma}_* \right),$$

with the marginal $q(x_*)$ being

$$q(x_*) = \mathcal{N}(x_*; \mathbf{k}_*^T\mathbf{K}^{-1}\boldsymbol{\mu}, \kappa_* - \mathbf{k}_*^T(\mathbf{K}+\tilde{\boldsymbol{\Sigma}})^{-1}\mathbf{k}_*) = \mathcal{N}(x_*; \mu_*, \sigma_*^2). \tag{34}$$

According to Equation (29), one needs the covariances Σ_{*j} to correct the marginal's mean; they appear in the last column of $\boldsymbol{\Sigma}_*$ in Equation (33). The correction is

$$\langle x_* \rangle_{p(\mathbf{x}, x_*)} - \mu_* = \sum_{l \geq 3} \sum_{j \neq n} \frac{\Sigma_{*j}}{\Sigma_{jj}} \frac{c_{l+1,j} c_{ln}}{l!} \left(\frac{\Sigma_{jn}}{\Sigma_{jj}\Sigma_{nn}} \right)^l + \cdots .$$

The sum over pairs $j \neq n$ include the added dimension $*$, and thus pairs $(j, *)$ and $(*, n)$. The cumulants for this problem, used both for EP and correcting it, are derived in Appendix E.2.

8.2.2 PREDICTIVE CORRECTIONS

In Figure 4 we investigate the predictive mean correction for two cases, one where the data cannot realistically be expected to appear under the prior, and the other where the prior is reasonable. For $s \in \mathbb{R}$, the values of $x(s_*)$ are predicted using a GP with squared exponential covariance function $k(s, s') = \theta \exp(-\frac{1}{2}(s-s')^2/\ell)$.

In the first instance, the prior amplitude θ and lengthscale ℓ are deliberately set to values that are too big; in other words, a typical sample from the prior would not match the observed data. We illustrate the posterior marginal $q(x_*)$, and using Equations (29) and (30), show visible corrections to its mean and variance.[4] For comparison, Figure 4 additionally shows what the predictive mean would have been were $\{y_n\}$ observed under Gaussian noise with the same mean and variance as $\mathcal{U}[-a, a]$: it is substantially different.

In the second instance, $\log Z_{\text{EP}}$ is maximized with respect to the covariance function hyperparameters θ and ℓ to get a kernel function that more reasonably describes the data. The correction

4. In the correction for the mean in Equation (29), we used $l = 3$ and $l = 4$ in the second order correction. For the correction to the variance in Equation (30), we used $l = 3$ in the first sum, and $l = 3$ and $l = 4$ in the second sum.

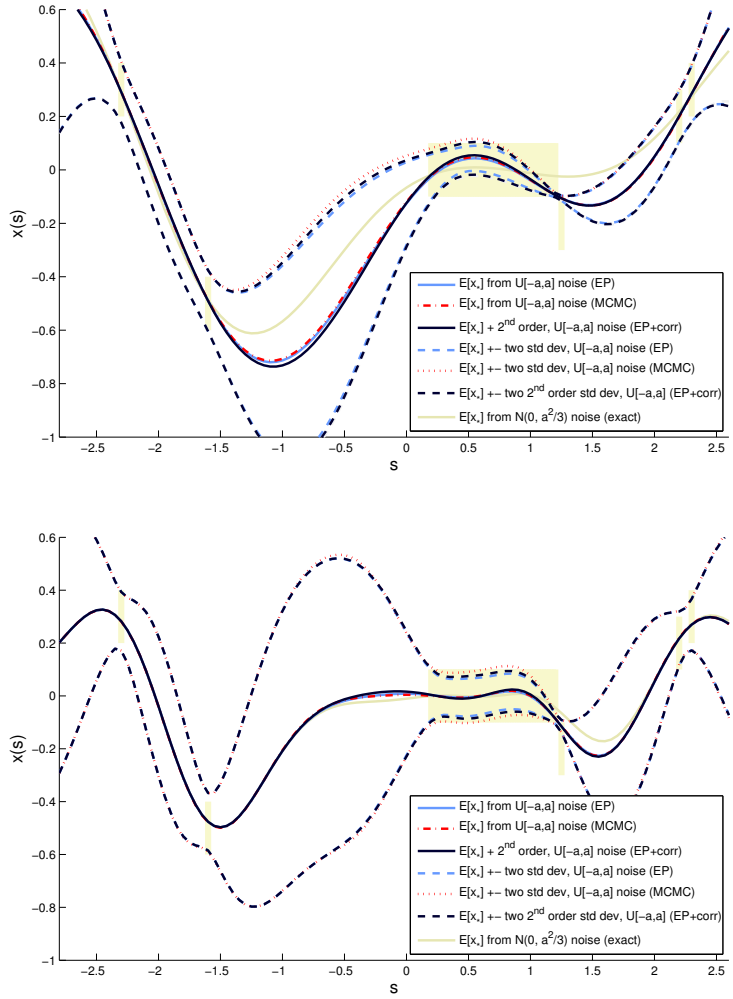


Figure 4: Predicting $x(s_*)$ with a GP. The "boxed" bars indicate the permissible $x(s_n)$ values; they are linked to observations y_n through the uniform likelihood $\mathbb{I}[|x_n - y_n| < a]$. Due to the $\mathcal{U}[-a,a]$ noise model, $q(x_*)$ is ambivalent to where in the "box" $x(s_*)$ is placed. A second order correction to the mean of $q(x_*)$ is shown in a dotted line. The lightly shaded function plots $p(x_*)$, *if* the likelihood was also Gaussian with variance matching that of the "box". In the *top* figure both the prior amplitude θ and lengthscale ℓ are overestimated. In the *bottom* figure, θ and ℓ were chosen by maximizing $\log Z_{\mathrm{EP}}$ with respect to their values. Notice the smaller EP approximation error.

to the mean of $q(s_*)$ is much smaller, and furthermore, generally follows the "Gaussian noise" posterior mean. When the observed data is not typical under the prior, the correction to $\langle x_* \rangle$ is substantially bigger than when the prior is representative of the data.

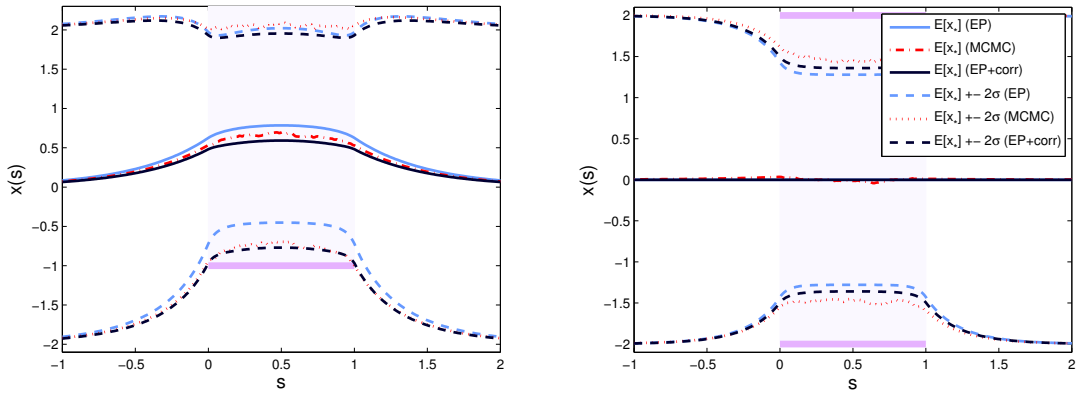


Figure 5: Predicting $x(s_*)$ with a GP with $k(s,s') = \exp\{-|s-s'|/2\ell\}$ and $\ell = 1$. In the *left* figure $\log R_{\text{MCMC}} = 0.41$, while the second order correction estimates it as $\log R \approx 0.64$. On the *right*, the correction to the variance is not as accurate as that on the *left*. The *right* correction is $\log R_{\text{MCMC}} = 0.28$, and its discrepancy with $\log R \approx 0.45$ (EP+corr) is much bigger.

8.2.3 UNDERESTIMATING THE TRUTH

Under closer inspection, the variance in Figure 4 is slightly underestimated in regions where there are many close box constraints $|x_n - y_n| < a$. However, under sparser constraints relative to the kernel width, EP accurately estimates the predictive mean and variance. In Figure 5 this is taken further: for $N = 100$ uniformly spaced inputs $s \in [0,1]$, it is clear that $q(\mathbf{x})$ becomes too narrow. The second order correction, on the other hand, provides a much closer estimate to the ground truth.

One might inquire about the behavior of the EP estimate as $N \to \infty$ in Figure 5. In the next section, this will be used as a basis for illustrating a special case where $\log Z_{\text{EP}}$ *diverges*.

8.3 Gaussian Process in a Box

In the following insightful example—a special case of uniform noise regression—$\log Z_{\text{EP}}$ diverges from the ground truth with more data. Consider the ratio of functions $x(s)$ over $[0,1]$, drawn from a GP prior with kernel $k(s,s')$, such that $x(s)$ lies within the $[-a,a]$ box. Figure 6 illustrates three random draws from a GP prior, two of which are not contained in the $[-a,a]$ interval. The ratio of functions contained in the interval is equal to the normalizing constant of

$$p(\mathbf{x}) = \frac{1}{Z}\prod_n \mathbb{I}\Big[|x_n| < a\Big]\,\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{K})\,. \tag{35}$$

The fraction of samples from the GP prior that lie inside $[-a,a]$ shouldn't change as the GP is sampled at increasing granularity of inputs s. As Figure 6 illustrates, the MCMC estimate of $\log Z$ converges to a constant as $N \to \infty$. The EP estimate $\log Z_{\text{EP}}$, on the other hand, diverges to $-\infty$. (The cumulants that are required for the correction in Equation (28), and recipes for deriving them, are given in Appendix E.1.) Of course the correction also depends on the value a chosen. Figure 7 shows that for both $a \to 0$ and $a \to \infty$ the correction is zero for large N.

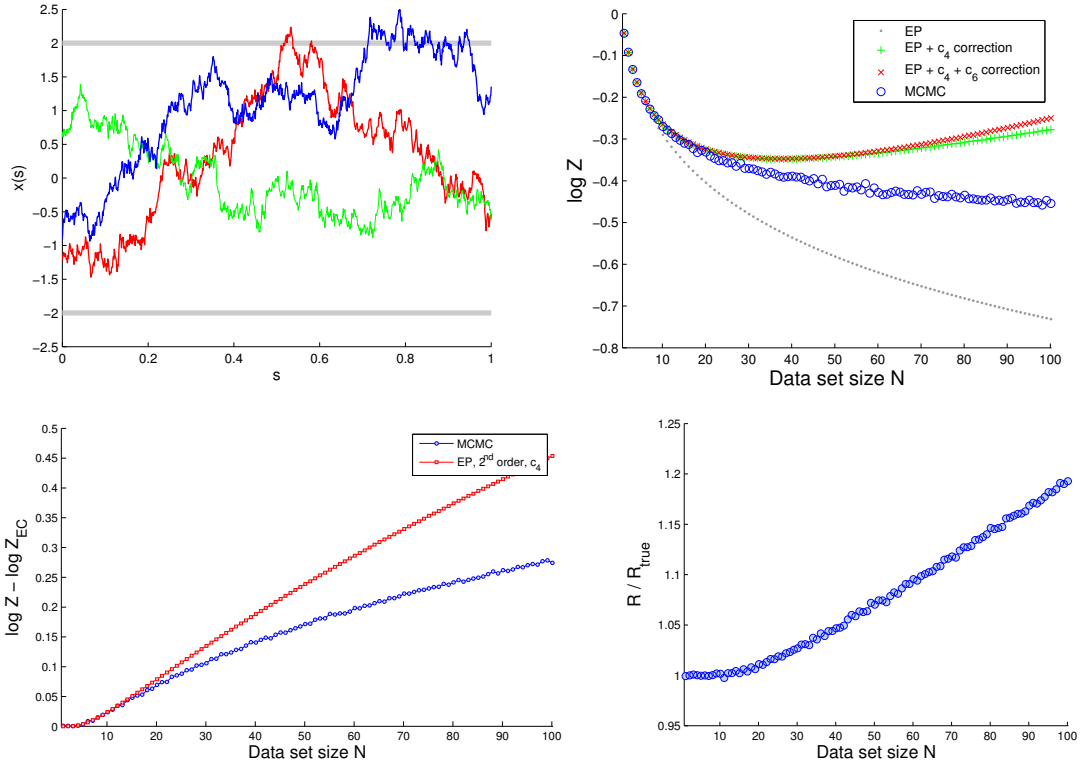


Figure 6: Samples from a GP prior with kernel $k(s,s') = \exp\{-|s-s'|/2\ell\}$ with $\ell = 1$, two of which are not contained in the $[-a,a]$ interval, are shown *top left*. As N increases in Equation (35), with $s_n \in [0,1]$, $\log Z_{\rm EP}$ diverges, while $\log Z$ converges to a constant. This is shown *top right*. The $+$'s and $\times$'s indicate the inclusion of the fourth ($+$) and fourth and sixth ($\times$) cumulants from the 2nd order in Equation (28) (an arrangement by total order would include 3rd order c_4–c_4–c_4 in $\times$). *Bottom left* and *right* show the growth for 2nd order c_4 correction relative to the exact correction.

An intuitive explanation, due to Philipp Hennig, takes a one-dimensional model $p(x) = \mathbb{I}[|x| < a]^N \mathcal{N}(x;0,1)$. A fully-factorized approximation therefore has $N-1$ *redundant* factors, as removing them doesn't change $p(x)$. However, each additional $\mathbb{I}[|x| < a]$ truncates the estimate, forcing EP to further reduce the variance of $q(x)$. The EP estimate using N factors $\mathbb{I}[|x| < a]^{1/N}$ is correct (see Appendix C for a similar example and analysis), even though the original problem remains unchanged. Even though this immediate solution cannot be applied to Equation (35), the *redundancy* across factors could be addressed by a principled junction tree-like factorization, where tuples of "neighboring" factors can be co-treated. Although beyond the scope of this paper, Appendix A gives a guideline on how to structure such an approximation.

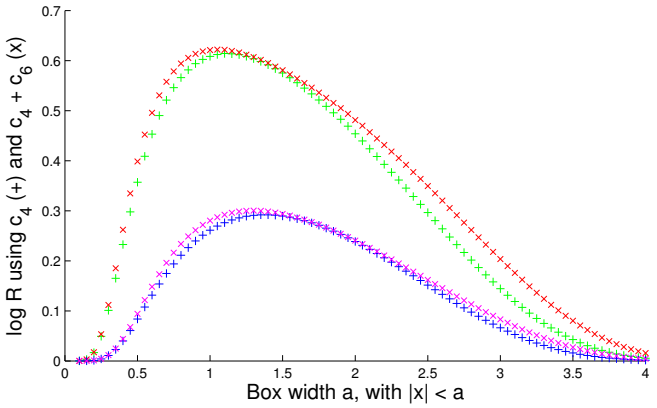


Figure 7: The accurateness of $\log Z_{\text{EP}}$ depends on the size of the $[-a,a]$ box relative to ℓ, with the estimation being exact as $a \to 0$ and $a \to \infty$. The second order correction for Figure 6's kernel is illustrated here over varying a's. The $+$'s and $\times$'s indicate the inclusion of the 4^{th} $(+)$ and 4^{th} and 6^{th} $(\times)$ cumulants in Equation (28). Of these, the top pair of lines are for $N = 100$, and the bottom pair for $N = 50$.

9. Ising Model Results

This section discusses various aspects of corrections to EP as applied to the Ising model—a Bayesian network with binary variables and pairwise potentials—in Equation (3).

We consider the set-up proposed by Wainwright and Jordan (2006) in which $N = 16$ nodes are either fully connected or connected to their nearest neighbors in a 4-by-4 grid. The external field (observation) strengths θ_i are drawn from a *uniform* distribution $\theta_i \sim \mathcal{U}[-d_{\text{obs}}, d_{\text{obs}}]$ with $d_{\text{obs}} = 0.25$. Three types of coupling strength statistics are considered: repulsive (anti-ferromagnetic) $J_{ij} \sim \mathcal{U}[-2d_{\text{coup}}, 0]$, mixed $J_{ij} \sim \mathcal{U}[-d_{\text{coup}}, +d_{\text{coup}}]$, and attractive (ferromagnetic) $J_{ij} \sim \mathcal{U}[0, +2d_{\text{coup}}]$.

Previously we have shown (Opper and Winther, 2005) that EP/EC gives very competitive results compared to several standard methods. In Section 9.1 we are interested in investigating whether a further improvement is obtained with the cumulant expansion. In Section 9.2, we revisit the correction approach proposed in Paquet et al. (2009) and make and empirical comparison with the cumulant approach.

9.1 Cumulant Expansion

For the factorized approximation we use Equations (26) and (29) for the $\log Z$ and marginal corrections, respectively. The expression for the cumulants of the Ising model is given in Section 4.2.1. The derivation of the corresponding tree expressions may be found in Appendices B and E.4.

Table 1 gives the average absolute deviation (AAD) of marginals

$$\text{AAD} = \frac{1}{N}\sum_i \Big| p(x_i = 1) - p(x_i = 1|\text{method}) \Big| = \frac{1}{2N}\sum_i \left| m_i - m_i^{\text{est}} \right| ,$$

while Table 2 gives the absolute deviation of $\log Z$ averaged of 100 repetitions. In two cases (Grid, $d_{\text{coup}} = 2$ Repulsive and Attractive coupling) we observed some numerical problems with the EC

Problem type			AAD marginals				
Graph	Coupling	d_{coup}	LBP	LD	EC	EC c	EC t
Full	Repulsive	0.25	.037	.020	.003	**.0006**	.0017
		0.50	.071	.018	.031	.0157	**.0143**
	Mixed	0.25	.004	.020	.002	**.0004**	.0013
		0.50	.055	.021	.022	.0159	**.0151**
	Attractive	0.06	.024	.027	.004	**.0023**	.0025
		0.12	.435	.033	.117	.1066	**.0211**
Grid	Repulsive	1.0	.294	.047	.153	*.1693*	**.0031**
		2.0	.342	.041	.198	*.4244*	**.0021**
	Mixed	1.0	.014	.016	.011	*.0122*	**.0018**
		2.0	.095	.038	.082	*.0984*	**.0068**
	Attractive	1.0	.440	.047	.125	*.1759*	**.0028**
		2.0	.520	.042	.177	*.4730*	**.0002**

Table 1: Average absolute deviation (AAD) of marginals in a Wainwright-Jordan set-up, comparing loopy belief propagation (LBP), log-determinant relaxation (LD), EC, EC with $l = 4$ second order correction (EC c), and an EC tree (EC t). Results in bold face highlight best results, while italics indicate where the cumulant expression is less accurate than the original approximation.

tree solver. It might be some cases that a solution does not exist but we ascribe numerical instabilities in our implementation as the main cause for these problems. It is currently out of the scope of this work to come up with a better solver. We choose to report the average performance for those runs that could attain a high degree of expectation consistency: $\sum_{i=1}^{N}(\langle x_i\rangle_{q_i} - \langle x_i\rangle_q)^2 \leq 10^{-20}$. This was 69 out of 100 in the mentioned cases and 100 of 100 in the remaining.

We observe that for the Grid simulations, the corrected marginals in factorized approximation are less accurate than the original approximation. In Figure 8 we vary the coupling strength for a specific set-up (Grid Mixed) and observe a cross-over between the correction and original for the error on marginals as the coupling strength increases. We conjecture that when the error of the original solution is high then the number of terms needed in the cumulant correction increases. The estimation of the marginal seems more sensitive to this than the $\log Z$ estimate. The tree approximation is very precise for the whole coupling strength interval considered and the fourth order cumulant in the second order expansion is therefore sufficient to get often quite large improvements over the original tree approximation.

9.2 The ε-Expansion

In Paquet et al. (2009) we introduced an alternative expansion for R and applied it to Gaussian processes and mixture models. It is obtained from Equation (12) using a finite series expansion, where the normalized deviation

$$\varepsilon_n(x_n) = \frac{q_n(x_n)}{q(x_n)} - 1$$

Problem type			Absolute deviation $\log Z$				
Graph	Coupling	d_{coup}	EC	EC c	EC εc	EC t	EC tc
Full	Repulsive	0.25	.0310	.0018	.0061	.0104	**.0010**
		0.50	.3358	.0639	.0697	.1412	**.0440**
	Mixed	0.25	.0235	.0013	.0046	.0129	**.0009**
		0.50	.3362	.0655	.0671	.1798	**.0620**
	Attractive	0.06	.0236	.0028	.0048	.0166	**.0006**
		0.12	.8297	**.1882**	.2281	.2672	.2094
Grid	Repulsive	1.0	1.7776	.8461	.8124	.0279	**.0115**
		2.0	4.3555	2.9239	3.4741	.0086	**.0077**
	Mixed	1.0	.3539	.1443	.0321	.0133	**.0039**
		2.0	1.2960	.7057	.4460	.0566	**.0179**
	Attractive	1.0	1.6114	.7916	.7546	.0282	**.0111**
		2.0	4.2861	2.9350	3.4638	.0441	**.0433**

Table 2: Absolute deviation log partition function in a Wainwright-Jordan set-up, comparing EC, EC with $l = 4$ second order correction (EC c), EC with a full second order ε expansion (EC εc), EC tree (EC t) and EC tree with $l = 4$ second order correction (EC tc). Results in bold face highlight best results. The cumulant expression is consistently more accurate than the original approximation.

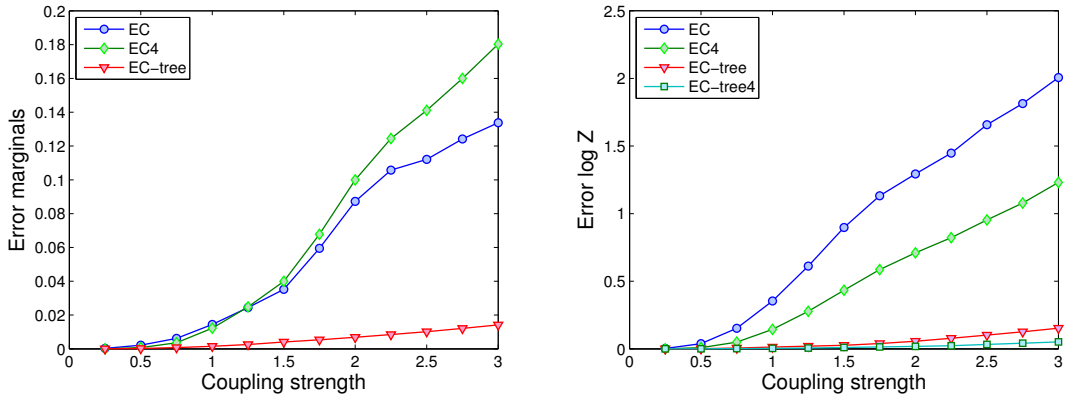


Figure 8: Error on marginal *(left)* and $\log Z$ *(right)* for grid and mixed couplings as a function of coupling strength.

is treated as the small quantity instead of higher order cumulants. R has an exact representation with 2^N terms that we may truncate at lowest non-trivial order:

$$R = \left\langle \prod_n (1+\varepsilon_n(x_n)) \right\rangle_{q(\mathbf{x})} \approx 1 + \sum_{m<n} \langle \varepsilon_m(x_m)\varepsilon_n(x_n)\rangle + O(\varepsilon^3) \; .$$

The linear terms are all equal to one because $\left\langle \frac{q_n(x_n)}{q(x_n)} \right\rangle_q = \int q(x_n) \frac{q_n(x_n)}{q(x_n)} \mathrm{d}x_n = 1$ and since $q_n(x_n)$ is a binary distribution the quadratic term becomes a weighted sum of ratios of Normal distributions:

$$\left\langle \frac{q_m(x_m)}{q(x_m)} \right\rangle_{q(\mathbf{x})} = \sum_{x_n, x_m = \pm 1} \frac{1 + x_m m_m}{2} \frac{1 + x_n m_n}{2} \frac{q(x_m, x_n)}{q(x_m) q(x_n)} \, .$$

The final expression for the lowest order approximation to R is then

$$R \approx 1 + \sum_{m<n} \sum_{x_n, x_m = \pm 1} \frac{1 + x_m m_m}{2} \frac{1 + x_n m_n}{2} \frac{q(x_m, x_n)}{q(x_m) q(x_n)} - \frac{N(N-1)}{2} \, .$$

From Table 2 we observe an improvement over the original factorized approximation and results similar to the cumulant correction to the factorized approximation for all settings. The ε-expansion may also used to calculate marginals and applied to generalized factorizations. These topics will be studied elsewhere.

10. Future Directions

Corrections to Gaussian EP approximations were examined in this paper. The Gaussian measure allowed for a convenient set of mathematical tools to be employed, mostly because it admits orthogonality of a set of polynomials, the Hermite polynomials, which allowed a clean simplification of many expressions. So far we have restricted ourselves to expansions to low orders in cumulants. Our results indicate that these first corrections to EP can already provide useful information about the quality of the EP solution. Small corrections typically show that EP is fairly accurate and the corrections improve on that. On the other hand, large corrections indicate that the EP approximation performs poorly. The low order corrections can yield a step in the right direction but in general their result may not be trusted and alternatives to the Gaussian EP approximation should be considered. It will be interesting to develop similar expansions to EP approximations with other exponential families besides the Gaussian one.

Can we expect that higher order terms in the cumulant expansion will give more reliable approximations? Before such a question could be attacked one first would need to decide in which order the terms of the expansion should be evaluated in order to obtain the most dominant contributions. For example, we might think of trying to first compute all terms in the second order expansion of the exponential in Equation (26), and then move on to higher orders. An alternative is to sort the expansion by the total sum of the orders of cumulants involved. This is in fact possible by introducing a suitable expansion parameter (which is later set equal to one) such that the formal Taylor series with respect to this parameter yields the desired expansion. However, it is not clear yet if and when such a power series expansion would actually converge. It may well be that our expansions are only of an asymptotic type (Boyd, 1999) for which the summation of only a certain number of terms might give an improvement whereas further terms would lead to worse results.

We expect that such questions could at least be answered for toy models such as the *Gaussian process in a box* model of Section 8.3. Our results for the latter example (together with the related *uniform noise regression* case) indicates that EP may not be understood as an off the shelf method for approximately calculating arbitrary high dimensional sums or integrals. One may conjecture that its quality strongly depends on the fact that such sums or integrals may or may not have an interpretation in terms of a proper statistical inference model which contain data that are highly

probable with respect to the model. It would be interesting to see if one can develop a theory for the average case performance of EP under such statistical assumptions of the data.

Appendix A. Factorizations: Gaussian Examples

As $p(\mathbf{x})$ is a latent Gaussian model, the g-terms in Equation (5) are chosen in this paper to give a Gaussian approximation

$$q(\mathbf{x}) = \frac{1}{Z_q} \exp\{\boldsymbol{\lambda}^T \phi(\mathbf{x})\} = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \ .$$

The sufficient statistics $\phi(\mathbf{x})$ and natural parameters $\boldsymbol{\lambda}$ of the Gaussian are defined as

$$\phi(\mathbf{x}) = (\mathbf{x}, -\tfrac{1}{2}\mathbf{x}\mathbf{x}^T) \quad \text{and} \quad \boldsymbol{\lambda} = (\boldsymbol{\gamma}, \boldsymbol{\Lambda}) \ ,$$

where $\boldsymbol{\lambda}^T \phi(\mathbf{x}) = \boldsymbol{\gamma}^T \mathbf{x} - \frac{1}{2} \mathrm{tr}[\boldsymbol{\Lambda}\mathbf{x}\mathbf{x}^T] = \boldsymbol{\gamma}^T \mathbf{x} - \frac{1}{2}\mathbf{x}^T \boldsymbol{\Lambda} \mathbf{x}$. There exists a bijection between the canonical parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ and natural parameters, such that the mean and covariance can be determined with $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}^{-1}$ and $\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\gamma}$.

In Equation (1) we can define $g_0(\mathbf{x}) = \exp\{\boldsymbol{\lambda}_0^T \phi(\mathbf{x})\}$, where $\boldsymbol{\lambda}_0 = (\boldsymbol{\gamma}^{(0)}, \boldsymbol{\Lambda}^{(0)})$, such that it is essentially a rescaling of factor f_0. In the Ising model in Equation (3), this means that $\boldsymbol{\Lambda}^{(0)} = -\mathbf{J}$ and $\boldsymbol{\gamma}^{(0)} = \boldsymbol{\theta}$. In the Gaussian process classification model in Equation (2), this implies that $\boldsymbol{\Lambda}^{(0)} = \mathbf{K}^{-1}$ and $\boldsymbol{\gamma}^{(0)} = \mathbf{0}$.

A.1 Term-Wise Factorizations

It remains to define a suitable factorization for the term-product $\prod_n t_n(x_n)$. This factorization can be fully factorized, factorized over disjoint sets of variables, factorized as a tree, or follow more arbitrary factorizations (see the simple example in Appendix C). A few such factorizations are given below in increasing orders of complexity. *In each case we do not include the f_0 factor for clarity.* Furthermore, even though the term factorization may be chosen to fully factorize, $q(\mathbf{x})$ may be fully connected through the inclusion of f_0.

A.1.1 FULLY FACTORIZED

A common factorization of $\prod_n t_n(x_n)$ is to set $f_n(\mathbf{x}) = t_n(x_n)$. The natural parameters of $g_n(\mathbf{x}) = \exp\{\boldsymbol{\lambda}_n^T \phi(\mathbf{x})\}$ are chosen to be $\boldsymbol{\lambda}_n = (\gamma_n^{(n)}, \Lambda_{nn}^{(n)})$, corresponding to $\phi_n(x_n) = (x_n, -\frac{1}{2}x_n^2)$. For clarity the other γ and Λ parameters in $\boldsymbol{\lambda}_n$ are not shown, as they are clamped at zero. This gives an approximation $q(\mathbf{x})$ that is defined by $\boldsymbol{\lambda} = \boldsymbol{\lambda}_0 + \sum_n \boldsymbol{\lambda}_n$.

A.1.2 FACTORIZATION INTO DISJOINT PAIRS

As a second step the N variables can be subdivided into *disjoint* pairs $\mathbf{x}_{\boldsymbol{\pi}} = (x_m, x_n)$. The factorization over terms couples pairs of variables through

$$\prod_n t_n(x_n) = \prod_{\boldsymbol{\pi}=(m,n)} [t_m(x_m) t_n(x_n)] = \prod_{\boldsymbol{\pi}} f_{\boldsymbol{\pi}}(\mathbf{x}) \ .$$

In this case each factor will have a contribution $g_{\boldsymbol{\pi}}(\mathbf{x}) = \exp\{\boldsymbol{\lambda}_{\boldsymbol{\pi}}^T \phi(\mathbf{x})\}$ to the overall approximation, and, as $g_{\boldsymbol{\pi}}$ is a function of two variables, it is parameterized by the "correlated Gaussian form" $\boldsymbol{\lambda}_{\boldsymbol{\pi}} = (\gamma_m^{(\boldsymbol{\pi})}, \gamma_n^{(\boldsymbol{\pi})}, \Lambda_{mm}^{(\boldsymbol{\pi})}, \Lambda_{nn}^{(\boldsymbol{\pi})}, \Lambda_{mn}^{(\boldsymbol{\pi})})$. By symmetry $\Lambda_{nm}^{(\boldsymbol{\pi})} = \Lambda_{mn}^{(\boldsymbol{\pi})}$. The resulting $q(\mathbf{x})$ is defined in terms of these disjoint sets with $\boldsymbol{\lambda} = \boldsymbol{\lambda}_0 + \sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{\boldsymbol{\pi}}$.

A.1.3 TREE-STRUCTURED FACTORIZATION

A tree structure factorization can be defined by extending the above "disjoint pairs" case to allow for overlaps between terms. Let $\mathcal{G}$ define a spanning tree structure over all $\mathbf{x}$, and let $\boldsymbol{\tau} = (m,n) \in \mathcal{G}$ define the edges in the tree. Let d_n be the number of edges emanating from node x_n in the graph. Through a clever regrouping of terms into a "junction tree" form with

$$\prod_n t_n(x_n) = \frac{\prod_{\boldsymbol{\tau}=(m,n)} [t_m(x_m) t_n(x_n)]}{\prod_n t_n(x_n)^{d_n-1}} = \frac{\prod_{\boldsymbol{\tau}} f_{\boldsymbol{\tau}}(\mathbf{x})}{\prod_n f_n(\mathbf{x})^{d_n-1}} ,$$

the term-approximation will be tree-structured. In this example the D_a powers are 1 for edge factors $f_{\boldsymbol{\tau}}$ and $(1-d_n)$ for node factors f_n. Let $g_{\boldsymbol{\tau}}(\mathbf{x})$ and $g_n(\mathbf{x})$ be parameterized by $\boldsymbol{\lambda}_{\boldsymbol{\tau}}$ and $\boldsymbol{\lambda}_n$, as was done in the two examples above. Using

$$\frac{\prod_{\boldsymbol{\tau}} g_{\boldsymbol{\tau}}(\mathbf{x})}{\prod_n g_n(\mathbf{x})^{d_n-1}} = \frac{\prod_{\boldsymbol{\tau}} \exp\{\boldsymbol{\lambda}_{\boldsymbol{\tau}}^T \phi(\mathbf{x})\}}{\prod_n \exp\{\boldsymbol{\lambda}_n^T \phi(\mathbf{x})\}^{d_n-1}} ,$$

the resulting $q(\mathbf{x})$ has parameter vector $\boldsymbol{\lambda} = \boldsymbol{\lambda}_0 + \sum_{\boldsymbol{\tau}} \boldsymbol{\lambda}_{\boldsymbol{\tau}} - \sum_n (d_n - 1)\boldsymbol{\lambda}_n$.

It is useful to note that the form of the tree-structured approximation given here is that used by Opper and Winther (2005); it approximates the "junction tree" form using a Power EP factorization (Minka, 2004). The factorization and stationary condition is *different* from that of Tree EP (Minka and Qi, 2004).

A.2 Stationary Point

The EP moment matching conditions from Equation (7) are uniquely met at the stationary point of $\log Z_{\mathrm{EP}}$ in Equation (8), and are shown here. Consider the logarithm of the normalizer,

$$\log Z_{\mathrm{EP}} = \log Z_q + \sum_a D_a \log Z_a . \tag{36}$$

Using the sufficient statistics and natural parameters defined above, the two normalizers that constitute Equation (36) are

$$\begin{aligned} Z_q &= \int \mathrm{e}^{\sum_a D_a \boldsymbol{\lambda}_a^T \phi(\mathbf{x})} \, \mathrm{d}\mathbf{x} , \\ Z_a &= \frac{1}{Z_q} \int \mathrm{e}^{\sum_b D_b \boldsymbol{\lambda}_b^T \phi(\mathbf{x}) - \boldsymbol{\lambda}_a^T \phi(\mathbf{x})} f_a(\mathbf{x}) \, \mathrm{d}\mathbf{x} . \end{aligned}$$

Using these definitions, the derivatives of the terms in Equation (36) with respect to some EP factor c's parameters $\boldsymbol{\lambda}_c$ are

$$\begin{aligned} \frac{\partial \log Z_q}{\partial \boldsymbol{\lambda}_c} &= D_c \langle \phi(\mathbf{x}) \rangle_q , \\ \frac{\partial \log Z_a}{\partial \boldsymbol{\lambda}_c} &= \begin{cases} D_c \langle \phi(\mathbf{x}) \rangle_{q_a} - D_c \langle \phi(\mathbf{x}) \rangle_q & \text{if } c \neq a \\ (D_c - 1) \langle \phi(\mathbf{x}) \rangle_{q_c} - D_c \langle \phi(\mathbf{x}) \rangle_q & \text{if } c = a \end{cases} . \end{aligned}$$

When $\partial \log Z_{\mathrm{EP}} / \partial \boldsymbol{\lambda}_c = \mathbf{0}$ for any c, the following therefore holds:

$$\mathbf{0} = (D_c - 1)(\langle \phi(\mathbf{x}) \rangle_{q_c} - \langle \phi(\mathbf{x}) \rangle_q) + \sum_{a \neq c} D_a (\langle \phi(\mathbf{x}) \rangle_{q_a} - \langle \phi(\mathbf{x}) \rangle_q) .$$

Let $\mathbf{D}$ be a square matrix where the values in column a are D_a; all the rows in $\mathbf{D}$ are equal and it is singular. Furthermore, let $\psi_a = \langle \phi(\mathbf{x}) \rangle_{q_a} - \langle \phi(\mathbf{x}) \rangle_q$. By stacking all the ψ_a's into a column vector ψ, the above set of equalities lead to a system of equations

$$\mathbf{0} = ((\mathbf{D} - \mathbf{I}) \otimes \mathbf{I}_{\text{dim}})\, \psi \ .$$

(The Kronecker product is only required as the sufficient statistics' differences ψ_a have dimensionality "dim", usually larger than one.) As $\mathbf{D} - \mathbf{I}$ is nonsingular, it is solved by $\psi = \mathbf{0}$, and hence $\langle \phi(\mathbf{x}) \rangle_{q_a} = \langle \phi(\mathbf{x}) \rangle_q$ for all a.

The choice of parameterization of $\boldsymbol{\lambda}_a$ might give an overcomplete representation, and the exact moment-matching conditions $\langle \phi(\mathbf{x}) \rangle_{q_a} = \langle \phi(\mathbf{x}) \rangle_q$ might have more than one unique solution. However, this does not invalidate that at the stationary point of Equation (36), all moment-matching conditions must hold.

Appendix B. Tree-Structured Approximation

Let the factorization of the term-product $\prod_n t_n(x_n)$ take the form of a tree $\mathcal{G}$ with edges $\boldsymbol{\tau} = (m,n) \in \mathcal{G}$, as is described in Appendix A.1.3. The number connections to a node or vertex n shall be denoted by d_n. From Equation (32) the second order expansion is

$$\begin{aligned} \log R = &\frac{1}{2} \sum_{\boldsymbol{\tau} \neq \boldsymbol{\tau}'} \langle \langle r_{\boldsymbol{\tau}} \rangle \langle r_{\boldsymbol{\tau}'} \rangle \rangle + \frac{1}{2} \sum_{m \neq n} (1-d_m)(1-d_n) \langle \langle r_m \rangle \langle r_n \rangle \rangle \\ &+ \sum_{\boldsymbol{\tau}, n} (1-d_n) \langle \langle r_{\boldsymbol{\tau}} \rangle \langle r_n \rangle \rangle + \frac{1}{2} \sum_n (1-d_n)(-d_n) \left\langle \langle r_n \rangle^2 \right\rangle + \cdots \ , \end{aligned} \tag{37}$$

where the inner expectations are over $\mathbf{k}_{\boldsymbol{\tau}} | \mathbf{x}$ and $k_n | \mathbf{x}$, while the outer expectations are over $\mathbf{x}$.[5] The edge-edge, edge-node, and node-node expectations that are needed in Equation (37) are given in the following three sections.

B.1 Edge-Edge Expectations

The edge-edge expectation provides a beautiful illustration of the combinatorics that may be involved in Wick's theorem. For $\boldsymbol{\tau} \neq \boldsymbol{\tau}'$, the following expectation needs to be evaluated:

$$\begin{aligned} &\langle \langle r_{\boldsymbol{\tau}}(\mathbf{k}_{\boldsymbol{\tau}}) \rangle \langle r_{\boldsymbol{\tau}'}(\mathbf{k}_{\boldsymbol{\tau}'}) \rangle \rangle \\ &\quad = \left\langle \sum_{l \geq 3} \sum_{s \geq 3} i^{l+s} \left\{ \sum_{|\boldsymbol{\alpha}| = l} \frac{c_{\boldsymbol{\alpha}\boldsymbol{\tau}}}{\boldsymbol{\alpha}!} \langle \mathbf{k}_{\boldsymbol{\tau}}^{\boldsymbol{\alpha}} \rangle_{\mathbf{k}_{\boldsymbol{\tau}} | \mathbf{x}} \right\} \left\{ \sum_{|\boldsymbol{\alpha}'| = s} \frac{c_{\boldsymbol{\alpha}'\boldsymbol{\tau}'}}{\boldsymbol{\alpha}'!} \left\langle \mathbf{k}_{\boldsymbol{\tau}'}^{\boldsymbol{\alpha}'} \right\rangle_{\mathbf{k}_{\boldsymbol{\tau}'} | \mathbf{x}} \right\} \right\rangle_{\mathbf{x}} \ . \end{aligned} \tag{38}$$

The vectors $\boldsymbol{\alpha}$ that are summed over to get $|\boldsymbol{\alpha}| = l$ are $\boldsymbol{\alpha} = (0,l), (1,l-1), \ldots, (l,0)$; let $\boldsymbol{\alpha} = (\alpha_1, l - \alpha_1)$ when $|\boldsymbol{\alpha}| = l$. From the independence of $\mathbf{k}_{\boldsymbol{\tau}} | \mathbf{x}$ and $\mathbf{k}_{\boldsymbol{\tau}'} | \mathbf{x}$,

$$\left\langle \langle \mathbf{k}_{\boldsymbol{\tau}}^{\boldsymbol{\alpha}} \rangle_{\mathbf{k}_{\boldsymbol{\tau}} | \mathbf{x}} \langle \mathbf{k}_{\boldsymbol{\tau}'}^{\boldsymbol{\alpha}'} \rangle_{\mathbf{k}_{\boldsymbol{\tau}'} | \mathbf{x}} \right\rangle_{\mathbf{x}} = \left\langle \langle \mathbf{k}_{\boldsymbol{\tau}}^{\boldsymbol{\alpha}} \mathbf{k}_{\boldsymbol{\tau}'}^{\boldsymbol{\alpha}'} \rangle_{\mathbf{k}_{\boldsymbol{\tau}}, \mathbf{k}_{\boldsymbol{\tau}'} | \mathbf{x}} \right\rangle_{\mathbf{x}} = \left\langle k_{\tau_1}^{\alpha_1} k_{\tau_2}^{l-\alpha_1} k_{\tau_1'}^{\alpha_1'} k_{\tau_2'}^{s-\alpha_1'} \right\rangle_{\mathbf{k}_{\boldsymbol{\tau}}, \mathbf{k}_{\boldsymbol{\tau}'}} \ , \tag{39}$$

5. Some readers might wonder why there is no $\frac{1}{2}$ associated with the sum over $(\boldsymbol{\tau}, n)$ in Equation (37). In the other quadratic sums, for example over $m \neq n$, each (m,n) pair appears twice, as $r_m r_n$ and as $r_n r_m$. Each edge-node pair makes only one appearance in the sum; if the sum double-counted by including node-edge pairs, a division by two would have been necessary.

and therefore $\langle\langle r_{\boldsymbol{\tau}}\rangle\langle r_{\boldsymbol{\tau}'}\rangle\rangle = \langle\langle r_{\boldsymbol{\tau}}\, r_{\boldsymbol{\tau}'}\rangle\rangle$ whenever $\boldsymbol{\tau} \neq \boldsymbol{\tau}'$.

Wick's theorem is again instrumental in computing $\langle \mathbf{k}_{\boldsymbol{\tau}}^{\boldsymbol{\alpha}} \mathbf{k}_{\boldsymbol{\tau}'}^{\boldsymbol{\alpha}'} \rangle$, as all possible pairings of the random variables $\mathbf{k}_{\boldsymbol{\tau}} = (k_{\tau_1}, k_{\tau_2})$ and $\mathbf{k}_{\boldsymbol{\tau}'} = (k_{\tau'_1}, k_{\tau'_2})$ need to be included. As $\langle k_{\tau_1}^2 \rangle = 0$, $\langle k_{\tau_1} k_{\tau_2} \rangle = 0$, $\langle k_{\tau'_1}^2 \rangle = 0$, and $\langle k_{\tau_1} k_{\tau'_2} \rangle = 0$, the only non-zero expectations in the Wick expansion of Equation (39) occur when *all* the variables in $\mathbf{k}_{\boldsymbol{\tau}}$ and $\mathbf{k}_{\boldsymbol{\tau}'}$ are paired. This immediately means that $\langle k_{\tau_1}^{\alpha_1} k_{\tau_2}^{l-\alpha_1} k_{\tau'_1}^{\alpha'_1} k_{\tau'_2}^{s-\alpha'_1} \rangle = 0$ whenever $l \neq s$, as there will be some remaining variables in $\mathbf{k}_{\boldsymbol{\tau}}$ (or $\mathbf{k}_{\boldsymbol{\tau}'}$) that can't be paired and have to be self-paired with zero expectation.

Given $l = s$, evaluate the expectation in Equation (39). We introduce the "pairing count" vector $\boldsymbol{\beta}$ with elements $\beta_j \in \mathbb{N}_0$ and constraint $\sum_{j=1}^4 \beta_j = l$. Let β_1 count the number of pairings of k_{τ_1} with $k_{\tau'_1}$, and β_2 count the number of pairings of k_{τ_1} with $k_{\tau'_2}$. As there are α_1 k_{τ_1} terms, the sum of its outgoing pairings should equal α_1 with

$$\beta_1 + \beta_2 = \alpha_1 \ .$$

A furthermore requirement is that

$$\beta_1 + \beta_3 = \alpha'_1 \ , \quad \beta_3 + \beta_4 = \alpha_2 \ , \quad \beta_2 + \beta_4 = \alpha'_2 \ ,$$

where $\alpha_2 = l - \alpha_1$ and $\alpha'_2 = l - \alpha'_1$, and β_3 and β_4 be as in the Wick expansion below. Define $\mathcal{B}$ to be the set of all such $\boldsymbol{\beta}$'s, and let $\mathcal{C}(\boldsymbol{\beta})$ count the number of permuted configurations for a given pairing $\boldsymbol{\beta}$. From Wick's theorem the expected value is equal to the sum over all possible pairings $\boldsymbol{\beta}$:

$$\left\langle k_{\tau_1}^{\alpha_1} k_{\tau_2}^{\alpha_2} k_{\tau'_1}^{\alpha'_1} k_{\tau'_2}^{\alpha'_2} \right\rangle_{\mathbf{k}_{\boldsymbol{\tau}}, \mathbf{k}_{\boldsymbol{\tau}'}} = \sum_{\boldsymbol{\beta} \in \mathcal{B}} \mathcal{C}(\boldsymbol{\beta}) \langle k_{\tau_1} k_{\tau'_1} \rangle^{\beta_1} \langle k_{\tau_1} k_{\tau'_2} \rangle^{\beta_2} \langle k_{\tau_2} k_{\tau'_1} \rangle^{\beta_3} \langle k_{\tau_2} k_{\tau'_2} \rangle^{\beta_4} \ .$$

A simple scheme to enumerate all $\boldsymbol{\beta} \in \mathcal{B}$ is to let

$$\boldsymbol{\beta} = \Big[\beta_1,\ \alpha_1 - \beta_1,\ \alpha'_1 - \beta_1,\ (l + \beta_1) - (\alpha_1 + \alpha'_1) \Big],$$

so that $\boldsymbol{\beta} \in \mathcal{B}$ for each $\beta_1 \in \{\max(0, (\alpha_1 + \alpha'_1) - l), \ldots, \min(\alpha_1, \alpha'_1)\}$. The remaining components of $\boldsymbol{\beta}$ are uniquely determined from β_1.

B.1.1 COUNTING PAIRINGS

How many permuted pairings $\mathcal{C}(\boldsymbol{\beta})$ are there?

1. There are $\binom{\alpha_1}{\beta_1}$ ways of choosing β_1 k_{τ_1}'s, and then $\frac{\alpha'_1!}{(\alpha'_1 - \beta_1)!}$ ways of choosing $k_{\tau'_1}$ to pair with.

2. This leaves a remaining $(\alpha_1 - \beta_1)$ k_{τ_1}'s, that need to be paired with $(l - \alpha'_1)$ $k_{\tau'_2}$'s. There are $\frac{(l-\alpha'_1)!}{((l-\alpha'_1)-(\alpha_1-\beta_1))!}$ such pairings.

3. There are also $\alpha'_1 - \beta_1$ remaining $k_{\tau'_1}$'s, that need to be paired with k_{τ_2} variables. There are $\binom{l-\alpha_1}{\alpha'_1-\beta_1}$ ways of picking a k_{τ_2}, and a further $(\alpha'_1 - \beta_1)!$ ways of arranging the remaining $k_{\tau'_1}$.

4. Finally, the $(l - \alpha'_1) - (\alpha_1 - \beta_1)$ remaining $k'_{\tau_2}s$ need to be coupled with the remaining $k_{\tau'_2}$'s, and there are $((l - \alpha'_1) - (\alpha_1 - \beta_1))!$ such arrangements.

Multiplying the possible pairings from the four steps above gives

$$\begin{aligned}\mathcal{C}(\boldsymbol{\beta}) &= \binom{\alpha_1}{\beta_1} \frac{\alpha_1'!}{(\alpha_1'-\beta_1)!} \frac{(l-\alpha_1')!}{((l+\beta_1)-(\alpha_1+\alpha_1'))!} \cdots \\ &\qquad \cdots \times \binom{l-\alpha_1}{\alpha_1'-\beta_1} (\alpha_1'-\beta_1)!\,((l+\beta_1)-(\alpha_1+\alpha_1'))! \\ &= \binom{\alpha_1}{\beta_1} \alpha_1'!\,(l-\alpha_1')! \binom{l-\alpha_1}{\alpha_1'-\beta_1},\end{aligned}$$

which adds up to the total number of possible pairings $\sum_{\boldsymbol{\beta}\in\mathcal{B}} \mathcal{C}(\boldsymbol{\beta}) = l!$. A further useful simplification is $\mathcal{C}(\boldsymbol{\beta})/\boldsymbol{\alpha}!\boldsymbol{\alpha}'! = 1/\boldsymbol{\beta}!$ when $|\boldsymbol{\alpha}| = |\boldsymbol{\alpha}'| = l$, and is used below.

B.1.2 EDGE-EDGE EXPECTATION

The absence of any self-interacting loops from Wick's theorem lets the $\sum_{s\geq 3}$ drop away in Equation (38), as all terms are zero except for when $l = s$. Substituting $\langle \mathbf{k}_{\boldsymbol{\tau}}^{\boldsymbol{\alpha}} \mathbf{k}_{\boldsymbol{\tau}'}^{\boldsymbol{\alpha}'} \rangle$ and $\mathcal{C}(\boldsymbol{\beta})$ into Equation (38) gives the final result,

$$\begin{aligned}&\big\langle \langle r_{\boldsymbol{\tau}}(\mathbf{k}_{\boldsymbol{\tau}}) \rangle \langle r_{\boldsymbol{\tau}'}(\mathbf{k}_{\boldsymbol{\tau}'}) \rangle \big\rangle \\ &\quad = \sum_{l\geq 3} (-1)^l \sum_{|\boldsymbol{\alpha}|=l} \sum_{|\boldsymbol{\alpha}'|=l} c_{\boldsymbol{\alpha}\boldsymbol{\tau}}\, c_{\boldsymbol{\alpha}'\boldsymbol{\tau}'} \left\{ \sum_{\boldsymbol{\beta}\in\mathcal{B}} \frac{1}{\boldsymbol{\beta}!} \langle k_{\tau_1} k_{\tau_1'} \rangle^{\beta_1} \langle k_{\tau_1} k_{\tau_2'} \rangle^{\beta_2} \langle k_{\tau_2} k_{\tau_1'} \rangle^{\beta_3} \langle k_{\tau_2} k_{\tau_2'} \rangle^{\beta_4} \right\}.\end{aligned}$$

B.2 Edge-Node Expectations

The derivation for the edge-node expectations is similar to that of the edge-edge case,

$$\begin{aligned}\big\langle \langle r_{\boldsymbol{\tau}}(\mathbf{k}_{\boldsymbol{\tau}}) \rangle \langle r_n(k_n) \rangle \big\rangle &= \left\langle \sum_{l\geq 3} \sum_{s\geq 3} i^{l+s} \sum_{|\boldsymbol{\alpha}|=l} \frac{c_{\boldsymbol{\alpha}\boldsymbol{\tau}}\, c_{sn}}{\boldsymbol{\alpha}!\, s!} \langle \mathbf{k}_{\boldsymbol{\tau}}^{\boldsymbol{\alpha}} \rangle_{\mathbf{k}_{\boldsymbol{\tau}}|\mathbf{x}} \langle k_n^s \rangle_{k_n|\mathbf{x}} \right\rangle_{\mathbf{x}} \\ &= \sum_{l\geq 3} (-1)^l \sum_{|\boldsymbol{\alpha}|=l} \frac{c_{\boldsymbol{\alpha}\boldsymbol{\tau}}\, c_{ln}}{\boldsymbol{\alpha}!} \langle k_{\tau_1} k_n \rangle^{\alpha_1} \langle k_{\tau_2} k_n \rangle^{l-\alpha_1},\end{aligned}$$

where the expectations in the last line are again over $\{\mathbf{k}_{\boldsymbol{\tau}}, k_n\}$. When $\langle \mathbf{k}_{\boldsymbol{\tau}}^{\boldsymbol{\alpha}} k_n^s \rangle$ is evaluated with Wick's theorem, there are α_1 copies of k_{τ_1}, $l-\alpha_1$ copies of k_{τ_2}, and s copies of k_n. The zero relation of $\mathbf{k}_{\boldsymbol{\tau}}$ and k_n ensures that the only non-zero terms in the Wick sum are those where all the k_τ's are paired with k_n's; in other words, when $l = s$. There are $l!$ possible pairings, which cancels $l!$ in the denominator.

The above edge-node expectation is for any edge and node in the tree, but notice that it simplifies greatly when the edge $\boldsymbol{\tau}$ is a connection to node n. Say τ_1 is the edge variable corresponding to x_n. In this case the covariance with respect to the *opposite* pair is zero, with $\langle k_{\tau_2}, k_n \rangle = 0$ (see Figure 2) and only *one* of the $\boldsymbol{\alpha}$'s will have a non-zero contribution to the sum, namely when $\boldsymbol{\alpha} = (l, 0)$.

B.3 Node-Node Expectations

The node-node expectation is given in Equation (27), and is also used for $\langle \langle r_n \rangle^2 \rangle$.[6]

6. Due to the square in $\langle \langle r_n \rangle^2_{k_n|\mathbf{x}} \rangle_{\mathbf{x}}$, the inner average $\langle r_n \rangle_{k_n|\mathbf{x}}$ should first be computed to give an expansion over Hermite polynomials in $x_n - \mu_n$. An example of such a result is given Appendix C. The orthogonality of these polynomials over $q(\mathbf{x}-\boldsymbol{\mu})$ allows $\langle \langle r_n \rangle^2_{k_n|\mathbf{x}} \rangle_{\mathbf{x}}$ to also reduce to Equation (27).

Appendix C. A Tractable, One-Dimensional Example

The following example illustrates a tractable one-dimensional model with two factors. It is shown analytically that the correction to $\log Z_{\rm EP}$ must be zero, and that the result is reflected in the higher-order terms in Equation (32), which are also zero.

Consider the factorization of a probit term with a Gaussian prior into

$$p(x) = \frac{1}{Z}\Phi(x)\,\mathcal{N}(x;0,1) = \frac{1}{Z} f_a(x)^{1/2} f_b(x)^{1/2}\mathcal{N}(x;0,1)\ ,$$

where $\Phi(x)$ is the cumulative Gaussian density function, and $f_a(x) = f_b(x) = \Phi(x)$. Z can be computed exactly, but for the sake of example $p(x)$ will be approximated with

$$q(x) = \frac{1}{Z_q} g_a(x)^{1/2} g_b(x)^{1/2}\mathcal{N}(x;0,1) = \mathcal{N}(x;\mu,\sigma^2)\ .$$

Choose $g_a(x) = \exp\{\phi(x)^T\boldsymbol{\lambda}_a\}$, and $g_b(x) = \exp\{\phi(x)^T\boldsymbol{\lambda}_b\}$. The q approximation has parameter vector $\boldsymbol{\lambda} = \boldsymbol{\lambda}_0 + \frac{1}{2}\boldsymbol{\lambda}_a + \frac{1}{2}\boldsymbol{\lambda}_b$. The EP fixed point is defined by $\boldsymbol{\lambda}_a = \boldsymbol{\lambda}_b$ and $Z_a = Z_b$. (For example, subtracting $\boldsymbol{\lambda}_a$ at the fixed point will leave $\boldsymbol{\lambda}_{\backslash a} = \boldsymbol{\lambda}_0 + \mathbf{0}$, which is equal to a scaled version of the prior $f_0(x)$. The factor $f_a(x) = \Phi(x)$ is hence incorporated into the prior, giving Z_a. By a symmetric argument, $Z_a = Z_b$.) Although it is trivial to show that $Z_{\rm EP} = Z_q Z_a^{1/2} Z_b^{1/2}$ will be equal to the true partition function Z, we shall prove it by showing that the correction term is $\log R = 0$.

C.1 Analytic Correction

In this section a transformation of variables from x to $y \sim \mathcal{N}(y;0,1)$, with $y = (x-\mu)/\sigma$, will be used to make the derivation slightly simpler, and therefore

$$k_a|y \sim \mathcal{N}\left(k_a\,; -\frac{iy}{\sigma}, \sigma^{-2}\right)\ , \qquad k_b|y \sim \mathcal{N}\left(k_b\,; -\frac{iy}{\sigma}, \sigma^{-2}\right)\ .$$

Below we analytically show that the correction $\log R$ is zero, and hence that

$$R = \left\langle \left\langle \mathrm{e}^{r_a(k_a)}\right\rangle_{k_a|y}^{1/2} \left\langle \mathrm{e}^{r_b(k_b)}\right\rangle_{k_b|y}^{1/2} \right\rangle_y = \left\langle \sqrt{\mathcal{F}_a(y)}\sqrt{\mathcal{F}_b(y)}\right\rangle_y = 1\ , \tag{40}$$

where $\mathcal{F}_a(y)$ is a shorthand for $\langle \mathrm{e}^{r_a(k_a)}\rangle_{k_a|y}$ and

$$r_a(k_a) = \sum_{l\geq 3} i^l \frac{c_{al}}{l!} k_a^l\ , \qquad r_b(k_b) = \sum_{l\geq 3} i^l \frac{c_{bl}}{l!} k_b^l\ .$$

Because $f_a = f_b$, the cumulants will be the same for all l, hence $c_{al} = c_{bl}$. Furthermore, $k_a|y$ and $k_b|y$ are both distributed according to the *same* density. Now define, using $\mathrm{e}^{r_a} = 1 + r_a + \frac{1}{2}r_a^2 + \cdots$,

$$\begin{aligned}
\mathcal{F}_a(y) &= \left\langle 1 + \sum_{l\geq 3} i^l \frac{c_{al}}{l!} k_a^l + \frac{1}{2}\sum_{l,s\geq 3} i^{l+s}\frac{c_{al}c_{as}}{l!s!}k_a^{l+s} + \cdots \right\rangle_{k_a|y} \\
&= \left\langle 1 + \sum_{l\geq 3}\frac{c_{al}}{l!}\left(\frac{1}{\sigma}\right)^l (y+iu)^l + \frac{1}{2}\sum_{l,s\geq 3}\frac{c_{al}c_{as}}{l!s!}\left(\frac{1}{\sigma}\right)^{l+s}(y+iu)^{l+s} + \cdots \right\rangle_u \\
&= 1 + \sum_{l\geq 3}\frac{c_{al}}{l!}\left(\frac{1}{\sigma}\right)^l H_l(y) + \frac{1}{2}\sum_{l,s\geq 3}\frac{c_{al}c_{as}}{l!s!}\left(\frac{1}{\sigma}\right)^{l+s} H_{l+s}(y) + \cdots
\end{aligned} \tag{41}$$

In the second line above a transformation of variables was made in the integral, with $u = \sigma k_a + iy$, such that $k_a = (u - iy)/\sigma$. The Jacobian $1/\sigma$ ensures proper normalization so that the average is over $u \sim \mathcal{N}(u; 0, 1)$. In the last line $H_l(y)$ is the *Hermite polynomial* of degree l,

$$\begin{aligned} &H_0(y) = 1\,, && H_1(y) = y\,, && H_2(y) = y^2 - 1\,, \\ &H_3(y) = y^3 - 3y\,, && H_4(x) = y^4 - 6y^2 + 3\,, && H_5(y) = y^5 - 10y^3 + 15y \quad \cdots \end{aligned}$$

which can be obtained for any real y and integer $l = 0, 1, 2, \ldots$ from the average $H_l(y) = \left\langle (y + iu)^l \right\rangle_u$ over $u \sim \mathcal{N}(u; 0, 1)$.[7]

The remarkable property $\langle H_l(y) \rangle_y = 0$ for all l, ensures that $\langle \mathcal{F}_a(y) \rangle_y = 1$ in Equation (41). Furthermore, $\mathcal{F}_a(y) = \mathcal{F}_b(y)$ follows from the equivalence in cumulants $c_{al} = c_{bl}$; the roots in Equation (40) disappear to give $\langle \mathcal{F}_a(y) \rangle_y$, proving that $R = 1$ in Equation (40).

C.2 Second Order Correction

The second order expansion in Equation (32) in Section 7 evaluates to zero, as the matching cumulants $c_{al} = c_{bl}$ and equal distributions of $k_a|x$ and $k_b|x$ ensure that $\langle r_a(k_a) \rangle_{k_a|x} = \langle r_b(k_b) \rangle_{k_b|x}$:

$$\begin{aligned} \log R &= \frac{1}{4} \left\langle \langle r_a(k_a) \rangle_{k_a|x} \langle r_b(k_b) \rangle_{k_b|x} \right\rangle_x - \frac{1}{8} \left(\left\langle \langle r_a(k_a) \rangle^2_{k_a|x} \right\rangle_x + \left\langle \langle r_b(k_b) \rangle^2_{k_b|x} \right\rangle_x \right) + \cdots \\ &= \frac{1}{4} \left\langle \langle r_a(k_a) \rangle^2_{k_a|x} \right\rangle_x - \frac{1}{8} \left(2 \left\langle \langle r_a(k_a) \rangle^2_{k_a|x} \right\rangle_x \right) + \cdots \\ &= 0 + \cdots \,. \end{aligned}$$

Appendix D. Corrections to Marginals Distributions

Corrections to the marginal distributions follow from a similar derivation to that of the normalizing constant. As a simplification, let the Gaussian approximation be centred with $\mathbf{y} = \mathbf{x} - \boldsymbol{\mu}$, so that $q(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \mathbf{0}, \boldsymbol{\Sigma})$, and assume that $q(\mathbf{x})$ is arises from the fully factorized approximation in Section 5. In this appendix corrections will be computed for the mean $\langle x_i - \mu_i \rangle_{p(\mathbf{x})} = \langle y_i \rangle_{p(\mathbf{y})}$, and variance $\left\langle (x_i - \mu_i)(x_j - \mu_j) - \Sigma_{ij} \right\rangle_{p(\mathbf{x})} = \left\langle y_i y_j \right\rangle_{p(\mathbf{y})} - \Sigma_{ij}$.

A further simplification that will be employed in the following section is a change of variables $\eta_n = k_n + i\Sigma^{-1}_{nn} y_n$, so that $\eta_n \sim \mathcal{N}(\eta_n; 0, \Sigma^{-1}_{nn})$. Let

$$z_n = \eta_n - i\Sigma^{-1}_{nn} y_n \,,$$

which is zero-mean complex Gaussian random variable with a relation $\left\langle z_n^2 \right\rangle = 0$ and $\langle z_m z_n \rangle = -\Sigma_{mn}/(\Sigma_{mm}\Sigma_{nn})$ when $m \neq n$. Following Equation (24), the correction reads

$$R = \left\langle \prod_n \left\langle r_n(k_n) \right\rangle_{k_n|y_n} \right\rangle_{\mathbf{y}} = \left\langle \prod_n \left\langle r_n\left(\eta_n - i\Sigma^{-1}_{nn} y_n\right) \right\rangle_{\eta_n} \right\rangle_{\mathbf{y}} = \left\langle \exp\left[\sum_n r_n(z_n) \right] \right\rangle_{\mathbf{z}} \,.$$

7. When $\mathcal{F}(y)$ in Equation (41) is rearranged as a power series in σ^l, we obtain an Edgeworth expansion to arbitrary order l. The deviation from the Gaussian $q(y)$ is thereby factorized out of tilted distribution with $q_a(y) = q(y)\mathcal{F}(y)$. The interested reader is pointed to Blinnikov and Moessner (1998).

D.1 The Marginal Mean

The lowest order correction to the EP marginal's mean follows from the result in Equation (13):

$$\begin{aligned}\langle y_i \rangle_{p(\mathbf{y})} &= \frac{1}{R}\left\langle y_i \, \mathrm{e}^{\sum_n r_n(z_n)} \right\rangle_{\mathbf{z}} \\ &= \frac{1}{R}\sum_j \Sigma_{ij}\left\langle \frac{\partial}{\partial y_j}\, \mathrm{e}^{\sum_n r_n(z_n)} \right\rangle \\ &= \frac{1}{R}\sum_j \Sigma_{ij}\left\langle \frac{\partial}{\partial y_j}\left(1+\sum_n r_n(z_n)+\frac{1}{2}\sum_{m,n} r_m(z_m) r_n(z_n)+\cdots\right)\right\rangle \\ &= \frac{1}{R}\sum_j \Sigma_{ij}\left\langle \frac{\partial r_j(z_j)}{\partial y_j}+\sum_n r_n(z_n)\frac{\partial r_j(z_j)}{\partial y_j}+\cdots\right\rangle .\end{aligned}$$

In the above expansion the first order term is $\frac{\partial r_j(z_j)}{\partial y_j} = \frac{\partial r_j(z_j)}{\partial z_j}\frac{\partial z_j}{\partial y_j} = -i\Sigma_{jj}^{-1}\frac{\partial r_j(z_j)}{\partial z_j}$, and disappears as $\left\langle \frac{\partial r_j(z_j)}{\partial z_j}\right\rangle = 0$. The $j=n$ second order term also disappears as $\left\langle r_j(z_j)\frac{\partial r_j(z_j)}{\partial z_j}\right\rangle = 0$. These equivalences can be seen by taking $r_j(z_j)$ (and also its derivative) as a expansion over powers of z_j; as $\langle z_j^2\rangle = 0$, Wick's theorem states that every expectation of powers of z_j should be zero. Hence

$$\langle y_i \rangle_{p(\mathbf{y})} = -\frac{i}{R}\sum_{j\neq n}\frac{\Sigma_{ij}}{\Sigma_{jj}}\left\langle r_n(z_n)\frac{\partial r_j(z_j)}{\partial z_j}\right\rangle_{\mathbf{z}} + \cdots . \tag{42}$$

The derivative of the characteristic function, as required in Equation (42), is

$$\frac{\partial r_j(z_j)}{\partial z_j} = \frac{\partial}{\partial z_j}\left[\sum_{l\geq 3} i^l \frac{c_{lj}}{l!} z_j^l\right] = i\sum_{l\geq 3} i^{l-1}\frac{c_{lj}}{(l-1)!} z_j^{l-1} = i\sum_{l\geq 2} i^l \frac{c_{l+1,j}}{l!} z_j^l .$$

The expectations for $j\neq n$ in Equation (42) evaluate to

$$\begin{aligned}\left\langle r_n(z_n)\frac{\partial r_j(z_j)}{\partial z_j}\right\rangle_{\mathbf{z}} &= i\sum_{s,l\geq 3} i^{s+l}\frac{c_{l+1,j}, c_{sn}}{l!s!}\left\langle z_j^l z_n^s\right\rangle + i\sum_{s\geq 3, l=2} i^{s+l}\frac{c_{l+1,j}}{2!s!}\left\langle z_j^l z_n^s\right\rangle \\ &= i\sum_{l\geq 3} i^{2l}\frac{c_{l+1,j}c_{ln}}{(l!)^2}\left\langle z_j^l z_n^l\right\rangle ,\end{aligned} \tag{43}$$

with the second term disappearing as $s > l = 2$ ensures that some z_n is always self-paired in Wick's theorem. Finally, by substituting Equation (43) into (42), the correction to the mean is

$$\langle y_i \rangle_{p(\mathbf{y})} = \sum_{l\geq 3}\sum_{j\neq n}\frac{\Sigma_{ij}}{\Sigma_{jj}}\frac{c_{l+1,j}c_{ln}}{l!}\left(\frac{\Sigma_{jn}}{\Sigma_{jj}\Sigma_{nn}}\right)^l \pm \cdots .$$

D.2 The Marginal Covariance

The correction to the second moments follow the same recipe as that of the marginal mean in Appendix D.1. We proceed by first treating y_i with

$$\begin{aligned}
\left\langle y_i y_j \right\rangle_{p(\mathbf{y})} &= \frac{1}{R} \left\langle y_i \left\{ y_j \, \mathrm{e}^{\sum_n r_n(z_n)} \right\} \right\rangle_{\mathbf{z}} \\
&= \frac{1}{R} \sum_k \Sigma_{ik} \left\langle \frac{\partial}{\partial y_k} \left\{ y_j \, \mathrm{e}^{\sum_n r_n(z_n)} \right\} \right\rangle \\
&= \frac{1}{R} \sum_k \Sigma_{ik} \left\langle \delta_{jk} \, \mathrm{e}^{\sum_n r_n(z_n)} + y_j \frac{\partial}{\partial y_k} \mathrm{e}^{\sum_n r_n(z_n)} \right\rangle \\
&= \Sigma_{ij} + \frac{1}{R} \sum_k \Sigma_{ik} \left\langle y_j \frac{\partial}{\partial y_k} \mathrm{e}^{\sum_n r_n(z_n)} \right\rangle \, .
\end{aligned}$$

Reapplying the recipe gives the correction to the covariance:

$$\begin{aligned}
\left\langle y_i y_j \right\rangle_{p(\mathbf{y})} - \Sigma_{ij} &= \frac{1}{R} \sum_{kl} \Sigma_{il} \Sigma_{jk} \left\langle \frac{\partial^2}{\partial y_k \partial y_l} \, \mathrm{e}^{\sum_n r_n(z_n)} \right\rangle_{\mathbf{z}} \\
&= -i \sum_{kl} \frac{\Sigma_{il}}{\Sigma_{ll}} \Sigma_{jk} \left\langle \frac{\partial}{\partial y_k} \frac{\partial r_l(z_l)}{\partial z_l} \mathrm{e}^{\sum_n r_n(z_n)} \right\rangle + \cdots \\
&= -\sum_{kl} \frac{\Sigma_{il}}{\Sigma_{ll}} \frac{\Sigma_{jk}}{\Sigma_{kk}} \left\langle \left[\delta_{kl} \frac{\partial^2 r_l(z_l)}{\partial z_l^2} + \frac{\partial r_k(z_k)}{\partial z_k} \frac{\partial r_l(z_l)}{\partial z_l} \right] \mathrm{e}^{\sum_n r_n(z_n)} \right\rangle \\
&= \sum_{s \geq 3} \sum_{k \neq l} \frac{\Sigma_{il} \Sigma_{jl}}{\Sigma_{ll}^2} \frac{c_{sk} c_{s+2,l}}{s!} \left(\frac{\Sigma_{kl}}{\Sigma_{kk} \Sigma_{ll}} \right)^s \\
&\qquad + \sum_{s \geq 3} \sum_{k \neq l} \frac{\Sigma_{il}}{\Sigma_{ll}} \frac{\Sigma_{jk}}{\Sigma_{kk}} \frac{c_{sk} c_{sl}}{s!} \left(\frac{\Sigma_{kl}}{\Sigma_{kk} \Sigma_{ll}} \right)^{s-1} + \cdots \, .
\end{aligned}$$

Appendix E. Higher Order Cumulants

Much of this paper hinges on cumulants beyond the second order. These are frequently more cumbersome to obtain than the initial moments that are required by EP. This appendix provides details of the cumulants used in this paper.

The cumulants of a distribution $q_n(x)$ can be obtained from its moments through

$$\begin{aligned}
c_3 &= \left\langle x^3 \right\rangle - 3 \left\langle x^2 \right\rangle \left\langle x \right\rangle + 2 \left\langle x \right\rangle^3 \, , \\
c_4 &= \left\langle x^4 \right\rangle - 4 \left\langle x^3 \right\rangle \left\langle x \right\rangle - 3 \left\langle x^2 \right\rangle^2 + 12 \left\langle x^2 \right\rangle \left\langle x \right\rangle^2 - 6 \left\langle x \right\rangle^4 \, , \\
c_5 &= \left\langle x^5 \right\rangle - 5 \left\langle x^4 \right\rangle \left\langle x \right\rangle - 10 \left\langle x^3 \right\rangle \left\langle x^2 \right\rangle + 20 \left\langle x^3 \right\rangle \left\langle x \right\rangle^2 + 30 \left\langle x^2 \right\rangle^2 \left\langle x \right\rangle - 60 \left\langle x^2 \right\rangle \left\langle x \right\rangle^3 + 24 \left\langle x \right\rangle^5 \, ;
\end{aligned}$$

they are derived for doubly-truncated Gaussian distributions in Appendices E.1 and E.2. One might also directly take derivatives of the cumulant generating function, and the cumulants of a Probit-times-Gaussian distribution, common to GP classification models, are derived this way in Appendix E.3.

The tree-structured approximation in Sections 7 and 9.1, and Appendices A.1.3 and B, require cumulants over two variables. They are presented in Appendix E.4 for the Ising model.

E.1 Doubly Truncated Centered Gaussian

Consider the centered distribution $q_n(x_n) \propto \mathbb{I}[|x_n| < a]\,\mathcal{N}(x_n; 0, \lambda_n^{-1})$. The odd moments of this tilted distributions are, by symmetry, $\langle x_n \rangle = \langle x_n^3 \rangle = \langle x_n^5 \rangle = 0$. Let

$$Z_n = 2\sqrt{\frac{\lambda}{2\pi}} \int_0^a \mathrm{e}^{-\frac{1}{2}\lambda x^2}\,\mathrm{d}x = 2\Phi(z) - 1\ , \qquad z = \sqrt{\lambda} a\ ,$$

with the Probit function being $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(z; 0, 1)\,\mathrm{d}z$. Subscripts n are dropped where they are clearly implied by their context. To get the even moments, consider

$$A_1 = \partial_\lambda \log Z_n = \partial_\lambda \log\left(\sqrt{\lambda}\int_{-a}^{a} dx\, \mathrm{e}^{-\frac{1}{2}\lambda x^2}\right) = \frac{1}{2\lambda} - \frac{1}{2}\left\langle x^2 \right\rangle\ ,$$
$$A_2 = \partial_\lambda^2 \log Z_n = -\frac{1}{2\lambda^2} + \frac{1}{4}\left(\left\langle x^4 \right\rangle - \left\langle x^2 \right\rangle^2\right)\ .$$

Using the partition function, we get

$$A_1 = \partial_\lambda \log\left(2\Phi(z) - 1\right) = \frac{a}{\sqrt{\lambda}}\left(\frac{\mathcal{N}(z)}{2\Phi(z)-1}\right)\ ,$$
$$A_2 = \frac{a^2}{2\lambda}\left(\frac{z\,\mathcal{N}(z)}{2\Phi(z)-1}\right) - \frac{a}{2\lambda^{3/2}}\left(\frac{\mathcal{N}(z)}{2\Phi(z)-1}\right) - \frac{a^2}{\lambda}\left(\frac{\mathcal{N}(z)}{2\Phi(z)-1}\right)^2\ ,$$

and thus

$$\left\langle x^2 \right\rangle = \frac{1}{\lambda} - 2A_1\ ,$$
$$\left\langle x^4 \right\rangle = \frac{2}{\lambda^2} + \left\langle x^2 \right\rangle^2 + 4A_2\ .$$

We can further determine $A_3 = \partial_\lambda^3 \log Z_n$ using the partition function, giving

$$A_3 = \frac{3a}{4\lambda^{5/2}}\left(\frac{\mathcal{N}(z)}{2\Phi(z)-1}\right) + \frac{3a^2}{4\lambda^2}\left(\frac{z\mathcal{N}(z)}{2\Phi(z)-1}\right) + \frac{3a^2}{2\lambda^2}\left(\frac{\mathcal{N}(z)}{2\Phi(z)-1}\right)^2 + \frac{2a^3}{\lambda^{3/2}}\left(\frac{\mathcal{N}(z)}{2\Phi(z)-1}\right)^3$$
$$+ \frac{3a^3}{2\lambda^{3/2}}\left(\frac{\mathcal{N}(z)}{2\Phi(z)-1}\right)\left(\frac{z\mathcal{N}(z)}{2\Phi(z)-1}\right) + \frac{a^3}{4\lambda^{3/2}}\left(\frac{(z^2-1)\mathcal{N}(z)}{2\Phi(z)-1}\right)\ .$$

Therefore

$$\left\langle x^6 \right\rangle = \frac{8}{\lambda^3} + \left\langle x^2 \right\rangle \left\langle x^4 \right\rangle - 2\left\langle x^2 \right\rangle^3 + 2\left\langle x^4 \right\rangle \left\langle x^2 \right\rangle - 8A_3\ .$$

E.2 Doubly Truncated Non-Centered Gaussian

The same calculation from Appendix E.1 can be repeated to get the moments of the non-centered truncated Gaussian $q_n(x_n) \propto \mathbb{I}[|x_n| < a]\,\mathcal{N}(x_n; \mu, \lambda_n^{-1})$. The subscripts n are dropped where evident. The partition function is

$$Z(\lambda, \mu) = \sqrt{\frac{\lambda}{2\pi}} \int_{-a}^{a} \mathrm{e}^{-\frac{1}{2}\lambda(x-\mu)^2}\,\mathrm{d}x = \Phi(z_{\max}) - \Phi(z_{\min})\ ,$$

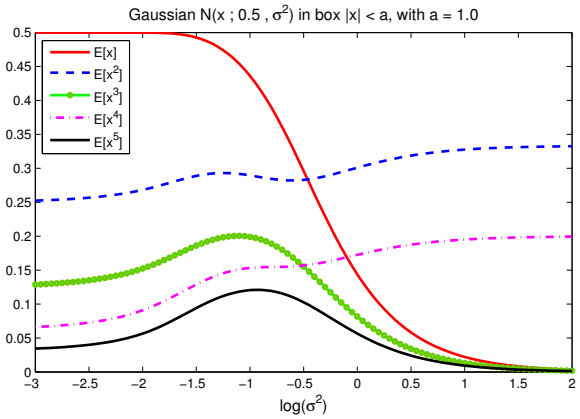


Figure 9: The moments of $q_n(x) \propto \mathbb{I}[|x| < a]\,\mathcal{N}(x;\mu,\sigma^2)$, as a function of σ^2. As the Gaussian variance $\sigma^2 \to \infty$, the moments converge to that of a uniform $\mathcal{U}[-a,a]$ distribution.

where

$$z_{\max} = \sqrt{\lambda}(\mu + a)\ , \quad z_{\min} = \sqrt{\lambda}(\mu - a)\ .$$

By again taking increasing derivatives of $Z(\lambda,\mu)$ with respect to μ and λ, the moments solved for are

$$\begin{aligned}
\langle x \rangle &= \mu + \frac{1}{\sqrt{\lambda}} \frac{\mathcal{N}(z_{\max}) - \mathcal{N}(z_{\min})}{\Phi(z_{\max}) - \Phi(z_{\min})}\ , \\
\langle x^2 \rangle &= 2\langle x \rangle \mu + \frac{1}{\lambda} - \mu^2 - \frac{1}{\lambda} \frac{z_{\max}\mathcal{N}(z_{\max}) - z_{\min}\mathcal{N}(z_{\min})}{\Phi(z_{\max}) - \Phi(z_{\min})}\ , \\
\langle x^3 \rangle &= 3\langle x^2 \rangle \mu + \langle x \rangle \left[\frac{3}{\lambda} - 3\mu^2\right] - \frac{3}{\lambda}\mu + \mu^3\ , \\
&\quad - \frac{1}{\lambda^{3/2}} \frac{(1 - z_{\max}^2)\,\mathcal{N}(z_{\max}) - (1 - z_{\min}^2)\,\mathcal{N}(z_{\min})}{\Phi(z_{\max}) - \Phi(z_{\min})}\ , \\
\langle x^4 \rangle &= 4\langle x^3 \rangle \mu + \langle x^2 \rangle \left[\frac{2}{\lambda} - 6\mu^2\right] + \langle x \rangle \left[4\mu^3 - \frac{4}{\lambda}\mu\right] + \frac{2}{\lambda}\mu^2 - \mu^4 + \frac{1}{\lambda^2} \\
&\quad - \frac{1}{\lambda^2} \frac{z_{\max}(1 + z_{\max}^2)\,\mathcal{N}(z_{\max}) - z_{\min}(1 + z_{\min}^2)\,\mathcal{N}(z_{\min})}{\Phi(z_{\max}) - \Phi(z_{\min})}\ .
\end{aligned}$$

Finally,

$$\begin{aligned}
\langle x^5 \rangle &= 5\langle x^4 \rangle \mu + \langle x^3 \rangle \left[\frac{6}{\lambda} - 10\mu^2\right] + \langle x^2 \rangle \left[10\mu^3 - \frac{18}{\lambda}\mu\right] + \langle x \rangle \left[\frac{18}{\lambda}\mu^2 - 5\mu^4 - \frac{3}{\lambda^2}\right] + \frac{3}{\lambda^2}\mu \\
&\quad - \frac{6}{\lambda}\mu^3 + \mu^5 - \frac{1}{\lambda^{5/2}} \frac{(1 + 2z_{\max}^2 - z_{\max}^4)\,\mathcal{N}(z_{\max}) - (1 + 2z_{\min}^2 - z_{\min}^4)\,\mathcal{N}(z_{\min})}{\Phi(z_{\max}) - \Phi(z_{\min})}\ .
\end{aligned}$$

As Figure 9 illustrates, these moments will converge to that of a uniform distribution as the Gaussian's variance grows large.

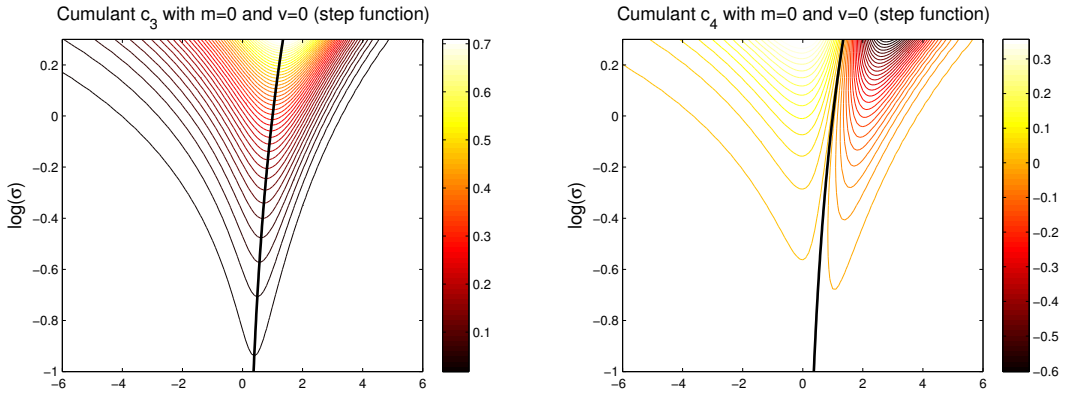


Figure 10: The third and fourth cumulants of the density $q_n(x) \propto \Phi((x-m)/v)\,\mathcal{N}(x;\mu,\sigma^2)$ in Appendix E.3. The step function $\Theta(x)$, with $m = v = 0$, is taken as an example here. The third cumulant is always positive, while the fourth cumulant is positive only when $\sigma > \mu$.

E.3 Probit Link Cumulants

EP approximations to Probit regression models, and Gaussian process classification models in general (see Section 8.1), depend on the moments of $q_n(x) \propto \Phi((x-m)/v)\,\mathcal{N}(x;\mu,\sigma^2)$. We introduce $v \geq 0$ so that the likelihood can become a step function at $v = 0$, for example. We shall obtain the cumulants by taking derivatives of the characteristic function. The characteristic function of $q_n(x)$, as described by Equation (15), is

$$\chi_n(k) = \left\langle e^{ikx} \right\rangle_{q_n(x)} = \exp\left\{ ik\mu - \frac{1}{2}k^2\sigma^2 \right\} \frac{\Phi(z_k)}{\Phi(z)} ,$$

with

$$z = \frac{\mu - m}{\sqrt{v^2+\sigma^2}} , \quad z_k = \frac{\mu + ik\sigma^2 - m}{\sqrt{v^2+\sigma^2}} .$$

The cumulants c_{ln} are determined from the derivatives of $\log \chi_n(k)$ at zero; a lengthy calculation shows that they are

$$c_{3n} = \alpha^3\beta\left[2\beta^2 + 3z\beta + z^2 - 1\right] ,$$
$$c_{4n} = -\alpha^4\beta\left[6\beta^3 + 12z\beta^2 + 7z^2\beta + z^3 - 4\beta - 3z\right] ,$$

where $\alpha = \sigma^2/\sqrt{v^2+\sigma^2}$ and $\beta = \mathcal{N}(z;0,1)/\Phi(z)$.

E.4 Two-Variable Ising Model Cumulants

We need some third and fourth order two-variable cumulants and thus generalize the results of Section 4.2 to the bivariate case. To do this we can exploit the cumulant generating property of $\log \chi_a(\mathbf{k}_a)$. Let $c_{(l,l')}$ denote the joint l, l' order cumulant of variable one and two, respectively. We can generate this cumulant from derivatives of $\log \chi_a(\mathbf{k}_a)$:

$$c_{(l,l')} = \left(\frac{\partial}{\partial ik_1}\right)^l \left(\frac{\partial}{\partial ik_2}\right)^{l'} \log \chi_a(\mathbf{k}_a)\Big|_{\mathbf{k}=\mathbf{0}} .$$

We can also express this as a recursion in terms of cumulants:

$$c_{(l+n,l'+n')} = \left(\frac{\partial}{\partial ik_1}\right)^n \left(\frac{\partial}{\partial ik_2}\right)^{n'} c_{(l,l')}(\mathbf{k})\bigg|_{\mathbf{k}=\mathbf{0}} .$$

By explicit calculation for a bivariate binary distribution we get the first two orders' cumulants: $c_{(1,0)} = m_1$, $c_{(0,1)} = m_2$, $c_{(2,0)} = 1 - m_1^2$, $c_{(0,2)} = 1 - m_2^2$ and $c_{(1,1)}$ is equal to the covariance between the two variables (to be matched with $q(x)$). The fact that we can write $c_{(2,0)}$ in terms of the first order cumulant shows that we can express all order cumulants in terms of the first and second order cumulant for example:

$$c_{(2,1)} = \frac{\partial}{\partial ik_2} c_{(2,0)}(\mathbf{k})\bigg|_{\mathbf{k}=\mathbf{0}} = \frac{\partial}{\partial ik_2}(1 - c_{(1,0)}^2(\mathbf{k}))\bigg|_{\mathbf{k}=\mathbf{0}} = -2c_{(1,0)}c_{(1,1)} .$$

Using the same recursion it is easy to show: $c_{(3,0)} = -2c_{(1,0)}c_{(2,0)}$, $c_{(4,0)} = -2c_{(2,0)}^2 - 2c_{(1,0)}c_{(3,0)}$, $c_{(3,1)} = -2c_{(2,0)}c_{(1,1)} - 2c_{(1,0)}c_{(2,1)}$ and $c_{(2,2)} = -2c_{(1,1)}^2 - 2c_{(1,0)}c_{(1,2)} = -2c_{(1,1)}^2 + 4c_{(1,0)}c_{(0,1)}c_{(1,1)}$.

References

D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

S. Blinnikov and R. Moessner. Expansions for nearly Gaussian distributions. *Astronomy and Astrophysics Supplement Series*, 130:193–205, 1998.

J. P. Boyd. The devil's invention: Asymptotic, superasymptotic and hyperasymptotic series. *Acta Applicandae Mathematicae*, 56:1–98, 1999.

M. Chertkov and V. Y. Chernyak. Loop series for discrete statistical models on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006:P06009, 2006.

B. Cseke and T. Heskes. Approximate marginals in latent Gaussian models. *Journal of Machine Learning Research*, 12:417–457, 2011.

M. Kuss and C. E. Rasmussen. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

M. Mézard, G. Parisi, and M. A. Virasoro. *Spin Glass Theory and Beyond*, volume 9 of *Lecture Notes in Physics*. World Scientific, 1987.

T. P. Minka. Expectation propagation for approximate Bayesian inference. In *UAI 2001*, pages 362–369, 2001a.

T. P. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT Media Lab, 2001b.

T. P. Minka. Power EP. Technical Report MSR-TR-2004-149, Microsoft Research Ltd, 2004.

T. P. Minka and Y. Qi. Tree-structured approximations by expectation propagation. In *Advances in Neural Information Processing Systems 16*. 2004.

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

M. Opper and O. Winther. Gaussian processes for classification: Mean field algorithms. *Neural Computation*, 12:2655–2684, 2000.

M. Opper and O. Winther. Expectation consistent approximate inference. *Journal of Machine Learning Research*, 6:2177–2204, 2005.

M. Opper, U. Paquet, and O. Winther. Improving on expectation propagation. In *Advances in Neural Information Processing Systems 21*, pages 1241–1248. 2009.

U. Paquet, M. Opper, and O. Winther. Perturbation corrections in approximate inference: Mixture modelling applications. *Journal of Machine Learning Research*, 10:935–976, 2009.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

H. Rue, S. Martino, and N. Chopin. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(2):319–392, 2009.

M. W. Seeger and H. Nickisch. Fast convergent algorithms for expectation propagation approximate Bayesian inference. *Arxiv preprint arXiv:1012.3584*, 2010.

D. Sherrington and S. Kirckpatrick. Solvable model of a spin-glass. *Phys. Rev. Lett.*, 35(26):1792–1796, December 1975.

E. Sudderth, M. Wainwright, and A. Willsky. Loop series and Bethe variational bounds in attractive graphical models. In *Advances in Neural Information Processing Systems 20*, pages 1425–1432. 2008.

D. J. Thouless, P. W. Anderson, and R. G. Palmer. Solution of a 'solvable model of a spin glass'. *Phil. Mag.*, 35:593, 1977.

M. A. J. van Gerven, B. Cseke, F. P. de Lange, and T. Heskes. Efficient Bayesian multivariate fMRI analysis using a sparsifying spatio-temporal prior. *NeuroImage*, 50(1):150–161, 2010.

M. J. Wainwright and M. I. Jordan. Log-determinant relaxation for approximate inference in discrete Markov random fields. *IEEE Transactions on Signal Processing*, 54(6):2099–2109, 2006.

M. Welling, A. Gelfand, and A. Ihler. A cluster-cumulant expansion at the fixed points of belief propagation. In *Uncertainty in Artificial Intelligence (UAI)*. 2012.

Journal of Machine Learning Research 14 (2013) 2899-2903 Submitted 11/12; Revised 4/13; Published 9/13

The CAM Software for Nonnegative Blind Source Separation in R-Java

Niya Wang WANGNY@VT.EDU
Fan Meng MENGFAN@VT.EDU
Department of Electrical and Computer Engineering
Virginia Polytechnic Institute and State University
Arlington, VA 22203, USA

Li Chen CHENL14@MAIL.NIH.GOV
Pediatric Oncology Branch, National Cancer Institute
National Institutes of Health
Gaithersburg, MD 20850, USA

Subha Madhavan SM696@GEORGETOWN.EDU
Innovation Center for Biomedical Informatics
Georgetown University
Washington DC 20007, USA

Robert Clarke CLARKER@GEORGETOWN.EDU
Lombardi Comprehensive Cancer Center
Georgetown University
Washington, DC 20057, USA

Eric P. Hoffman EHOFFMAN@CNMCRESEARCH.ORG
Research Center for Genetic Medicine
Children's National Medical Center
Washington, DC 20010, USA

Jianhua Xuan XUAN@VT.EDU
Yue Wang YUEWANG@VT.EDU
Department of Electrical and Computer Engineering
Virginia Polytechnic Institute and State University
Arlington, VA 22203, USA

Editor: Antti Honkela

Abstract

We describe a R-Java CAM (convex analysis of mixtures) package that provides comprehensive analytic functions and a graphic user interface (GUI) for blindly separating mixed nonnegative sources. This open-source multiplatform software implements recent and classic algorithms in the literature including Chan et al. (2008), Wang et al. (2010), Chen et al. (2011a) and Chen et al. (2011b). The CAM package offers several attractive features: (1) instead of using proprietary MATLAB, its analytic functions are written in R, which makes the codes more portable and easier to modify; (2) besides producing and plotting results in R, it also provides a Java GUI for automatic progress update and convenient visual monitoring; (3) multi-thread interactions between the R and Java modules are driven and integrated by a Java GUI, assuring that the whole CAM software runs responsively; (4) the package offers a simple mechanism to allow others to plug-in additional R-functions.

Keywords: convex analysis of mixtures, blind source separation, affinity propagation clustering, compartment modeling, information-based model selection

1. Overview

Blind source separation (BSS) has proven to be a powerful and widely-applicable tool for the analysis and interpretation of composite patterns in engineering and science (Hillman and Moore, 2007; Lee and Seung, 1999). BSS is often described by a linear latent variable model **X = AS**, where **X** is the observation data matrix, **A** is the unknown mixing matrix, and **S** is the unknown source data matrix. The fundamental objective of BSS is to estimate both the unknown but informative mixing proportions and the source signals based only on the observed mixtures (Child, 2006; Cruces-Alvarez et al., 2004; Hyvarinen et al., 2001; Keshava and Mustard, 2002).

While many existing BSS algorithms can usefully extract interesting patterns from mixture observations, they often prove inaccurate or even incorrect in the face of real-world BSS problems in which the pre-imposed assumptions may be invalid. There is a family of approaches exploiting the source non-negativity, including the non-negative matrix factorization (NMF) (Gillis, 2012; Lee and Seung, 1999). This motivates the development of alternative BSS techniques involving exploitation of source nonnegative nature (Chan et al., 2008; Chen et al., 2011a,b; Wang et al., 2010). The method works by performing convex analysis of mixtures (CAM) that automatically identifies pure-source signals that reside at the vertices of the multifaceted simplex most tightly enclosing the data scatter, enabling geometrically-principled delineation of distinct source patterns from mixtures, with the number of underlying sources being suggested by the minimum description length criterion.

Consider a latent variable model $\mathbf{x}(i) = \mathbf{A}\mathbf{s}(i)$, where the observation vector $\mathbf{x}(i) = [x_1(i), ..., x_M(i)]^T$ can be expressed as a non-negative linear combination of the source vectors $\mathbf{s}(i) = [s_1(i), ..., s_J(i)]^T$, and $\mathbf{A} = [\mathbf{a}_1, ..., \mathbf{a}_J]$ is the mixing matrix with $\mathbf{a}_j$ being the jth column vector. This falls neatly within the definition of a convex set (Fig. 1) (Chen et al., 2011a):

$$\mathbf{X} = \left\{ \sum_{j=1}^{J} s_j(i)\mathbf{a}_j | \mathbf{a}_j \in \mathbf{A}, s_j(i) \geq 0, \sum_{j=1}^{J} s_j(i) = 1, i = 1, ..., N \right\}.$$

Assume that the sources have at least one sample point whose signal is exclusively enriched in a particular source (Wang et al., 2010), we have shown that the vertex points of the observation simplex (Fig. 1) correspond to the column vectors of the mixing matrix (Chen et al., 2011b). Via a minimum-error-margin volume maximization, `CAM` identifies the optimum set of the vertices (Chen et al., 2011b; Wang et al., 2010). Using the samples attached to the vertices, compartment modeling (CM) (Chen et al., 2011a) obtains a parametric solution of **A**, nonnegative independent component analysis (nICA) (Oja and Plumbley, 2004) estimates **A** (and **s**) that maximizes the independency in **s**, and nonnegative well-grounded component analysis (nWCA) (Wang et al., 2010) finds the column vectors of **A** directly from the vertex cluster centers.

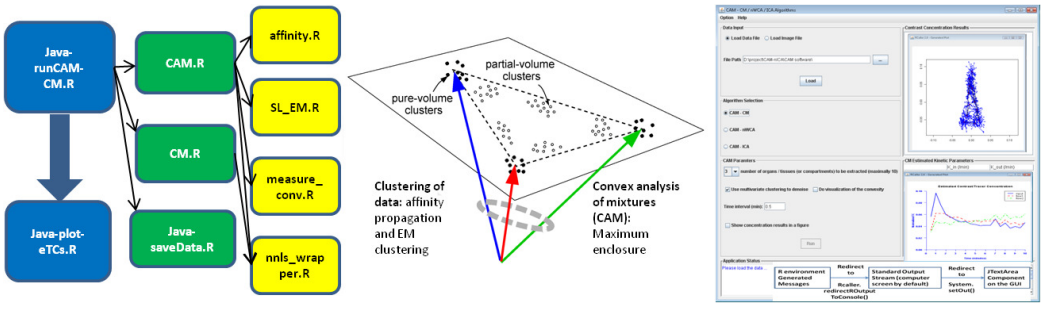


Figure 1: Schematic and illustrative flowchart of `R-Java CAM` package.

In this paper we describe a newly developed R-Java CAM package whose analytic functions are written in R, while a graphic user interface (GUI) is implemented in Java, taking full advantages of both programming languages. The core software suite implements CAM functions and includes normalization, clustering, and data visualization. Multi-thread interactions between the R and Java modules are driven and integrated by a Java GUI, which not only provides convenient data or parameter passing and visual progress monitoring but also assures the responsive execution of the entire CAM software.

2. Software Design and Implementation

The CAM package mainly consists of R and Java modules. The R module is a collection of *main* and *helper* functions, each represented by an R function object and achieving an independent and specific task (Fig. 1). The R module mainly performs various analytic tasks required by CAM: figure plotting, update, or error message generation. The Java module is developed to provide a GUI (Fig. 2). We adopt the model-view-controller (MVC) design strategy, and use different Java classes to separately perform information visualization and human-computer interaction. The Java module also serves as the *software driver and integrator* that use a multi-thread strategy to facilitate the interactions between the R and Java modules, such as importing raw data, passing algorithmic parameters, calling R scripts, and transporting results and messages.

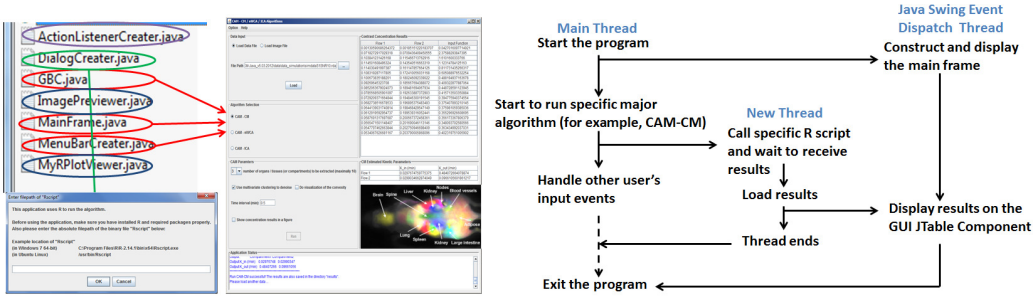


Figure 2: Interactive Java GUI supported by a multi-thread design strategy.

2.1 Analytic and Presentation Tasks Implemented in R

The R module performs the CAM algorithm and facilitates a suite of subsequent analyses including CM, nICA, and nWCA. These tasks are performed by the three *main* functions: CAM-CM.R, CAM-nICA.R, and CAM-nWCA.R, which can be activated by the three R scripts: Java-runCAM-CM.R, Java-runCAM-ICA.R, and Java-runCAM-nWCA.R. The R module also performs auxiliary tasks including automatic R library installation, figure drawing, and result recording; and offers other standard methods such as nonnegative matrix factorization (Lee and Seung, 1999), Fast ICA (Hyvarinen et al., 2001), factor analysis (Child, 2006), principal component analysis, affinity propagation, k-means clustering, and expectation-maximization algorithm for learning standard finite normal mixture model.

2.2 Graphic User Interface Written in Java Swing

The Java GUI module allows users to import data, select algorithms and parameters, and display results. The module encloses two packages: guiView contains classes for handling frames and

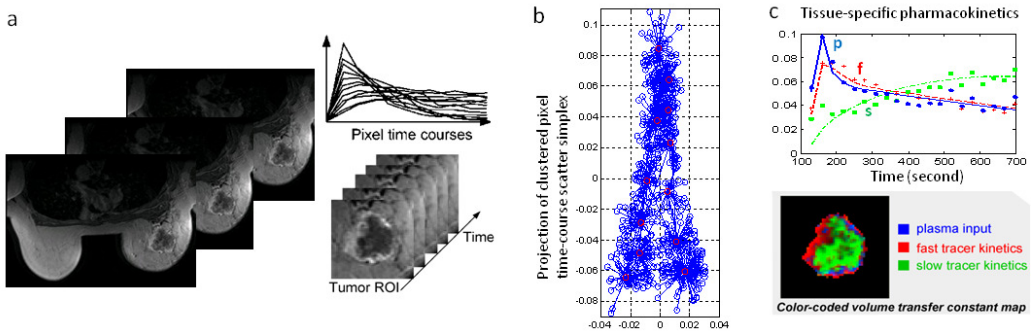


Figure 3: Application of `R-Java CAM` to deconvolving dynamic medical image sequence.

dialogs for managing user inputs; `guiModel` contains classes for representing result data sets and for interacting with the `R` script caller. Packaged as one jar file, the `GUI` module runs automatically.

2.3 Functional Interaction Between `R` and `Java`

We adopt the open-source program `RCaller (http://code.google.com/p/rcaller)` to implement the interaction between `R` and `Java` modules (Fig. 2), supported by explicitly designed `R` scripts such as `Java-runCAM-CM.R`. Specifically, five featured `Java` classes are introduced to interact with `R` for importing data or parameters, running algorithms, passing on or recording results, displaying figures, and handing over error messages. The examples of these classes include guiModel.MyRCaller.java, guiModel.MyRCaller.readResults(), and guiView.MyRPlotViewer.

3. Case Studies and Experimental Results

The `CAM` package has been successfully applied to various data types. Using dynamic contrast-enhanced magnetic resonance imaging data set of an advanced breast cancer case (Chen, et al., 2011b),"double click" (or command lines under Ubuntu) activated execution of `CAM-Java.jar` reveals two biologically interpretable vascular compartments with distinct kinetic patterns: fast clearance in the peripheral "rim" and slow clearance in the inner "core". These outcomes are consistent with previously reported intratumor heterogeneity (Fig. 3). Angiogenesis is essential to tumor development beyond 1-2mm^3. It has been widely observed that active angiogenesis is often observed in advanced breast tumors occurring in the peripheral "rim" with co-occurrence of inner-core hypoxia. This pattern is largely due to the defective endothelial barrier function and outgrowth blood supply. In another application to natural image mixtures, `CAM` algorithm successfully recovered the source images in a large number of trials (see Users Manual).

4. Summary and Acknowledgements

We have developed a `R-Java CAM` package for blindly separating mixed nonnegative sources. The open-source cross-platform software is easy-to-use and effective, validated in several real-world applications leading to plausible scientific discoveries. The software is freely downloadable from `http://mloss.org/software/view/437/`. We intend to maintain and support this package in the future. This work was supported in part by the US National Institutes of Health under Grants CA109872, CA 100970, and NS29525. We thank T.H. Chan, F.Y. Wang, Y. Zhu, and D.J. Miller for technical discussions.

References

T.H. Chan, W.K. Ma, C.Y. Chi, and Y. Wang. A convex analysis framework for blind separation of non-negative sources. *IEEE Transactions on Signal Processing*, 56:5120–5143, 2008.

L. Chen, T.H. Chan, P.L. Choyke, and E.M. Hillman et al. Cam-cm: a signal deconvolution tool for in vivo dynamic contrast-enhanced imaging of complex tissues. *Bioinformatics*, 27:2607–2609, 2011a.

L. Chen, P.L. Choyke, T.H. Chan, and C.Y. Chi et al. Tissue-specific compartmental analysis for dynamic contrast-enhanced mr imaging of complex tumors. *IEEE Transactions on Medical Imaging*, 30:2044–2058, 2011b.

D. Child. The essentials of factor analysis. *Continuum International*, 2006.

S.A. Cruces-Alvarez, Andrzej Cichocki, and Shun ichi Amari. From blind signal extraction to blind instantaneous signal separation: criteria, algorithms, and stability. *IEEE Transactions on Neural Networks*, 15:859–873, 2004.

N. Gillis. Sparse and unique nonnegative matrix factorization through data preprocessing. *Journal of Machine Learning Research*, 13:3349–3386, 2012.

E.M.C. Hillman and A. Moore. All-optical anatomical co-registration for molecular imaging of small animals using dynamic contrast. *Nature Photonics*, 1:526–530, 2007.

A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley, New York, 2001.

N. Keshava and J.F. Mustard. Spectral unmixing. *IEEE Signal Processing Magazine*, 19:44–57, 2002.

D.D. Lee and H.S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

E. Oja and M. Plumbley. Blind separation of positive sources by globally convergent gradient search. *Neural Computation*, 16:1811–1825, 2004.

F.Y. Wang, C.Y. Chi, T.H. Chan, and Y. Wang. Nonnegative least-correlated component analysis for separation of dependent sources by volume maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:857–888, 2010.

Journal of Machine Learning Research 14 (2013) 2905-2943 Submitted 6/12; Revised 5/13; Published 9/13

A Near-Optimal Algorithm for Differentially-Private Principal Components*

Kamalika Chaudhuri KCHAUDHURI@UCSD.EDU
Department of Computer Science and Engineering
University of California, San Diego
9500 Gilman Drive MC 0404
La Jolla, CA, 92093-0404, USA

Anand D. Sarwate ASARWATE@ALUM.MIT.EDU
Toyota Technological Institute at Chicago
6045 S. Kenwood Ave
Chicago, IL 60637, USA

Kaushik Sinha KAUSHIK.SINHA@WICHITA.EDU
Department of Electrical Engineering and Computer Science
Wichita State University
1845 Fairmount (Campus Box 83)
Wichita, KS 67260-0083, USA

Editor: Gabor Lugosi

Abstract

The principal components analysis (PCA) algorithm is a standard tool for identifying good low-dimensional approximations to high-dimensional data. Many data sets of interest contain private or sensitive information about individuals. Algorithms which operate on such data should be sensitive to the privacy risks in publishing their outputs. Differential privacy is a framework for developing tradeoffs between privacy and the utility of these outputs. In this paper we investigate the theory and empirical performance of differentially private approximations to PCA and propose a new method which explicitly optimizes the utility of the output. We show that the sample complexity of the proposed method differs from the existing procedure in the scaling with the data dimension, and that our method is nearly optimal in terms of this scaling. We furthermore illustrate our results, showing that on real data there is a large performance gap between the existing method and our method.

Keywords: differential privacy, principal components analysis, dimension reduction

1. Introduction

Dimensionality reduction is a fundamental tool for understanding complex data sets that arise in contemporary machine learning and data mining applications. Even though a single data point can be represented by hundreds or even thousands of features, the phenomena of interest are often intrinsically low-dimensional. By reducing the "extrinsic" dimension of the data to its "intrinsic" dimension, analysts can discover important structural relationships between features, more efficiently

*. A preliminary version of this work appeared at the Neural Information Processing Systems conference (Chaudhuri et al., 2012). This full version contains more experimental details, full proofs, and additional discussion.

use the transformed data for learning tasks such as classification or regression, and greatly reduce the space required to store the data. One of the oldest and most classical methods for dimensionality reduction is principal components analysis (PCA), which computes a low-rank approximation to the second moment matrix A of a set of points in $\mathbb{R}^d$. The rank k of the approximation is chosen to be the intrinsic dimension of the data. We view this procedure as specifying a k-dimensional subspace of $\mathbb{R}^d$.

Much of today's machine-learning is performed on the vast amounts of personal information collected by private companies and government agencies about individuals: examples include user or customer behaviors, demographic surveys, and test results from experimental subjects or patients. These data sets contain sensitive information about individuals and typically involve a large number of features. It is therefore important to design machine-learning algorithms which discover important structural relationships in the data while taking into account its sensitive nature. We study approximations to PCA which guarantee differential privacy, a cryptographically motivated definition of privacy (Dwork et al., 2006b) that has gained significant attention over the past few years in the machine-learning and data-mining communities (Machanavajjhala et al., 2008; McSherry and Mironov, 2009; McSherry, 2009; Friedman and Schuster, 2010; Mohammed et al., 2011). Differential privacy measures privacy risk by a parameter ε_p that bounds the log-likelihood ratio of output of a (private) algorithm under two databases differing in a single individual.

There are many general tools for providing differential privacy. The sensitivity method due to Dwork et al. (2006b) computes the desired algorithm (in our case, PCA) on the data and then adds noise proportional to the maximum change than can be induced by changing a single point in the data set. The PCA algorithm is very sensitive in this sense because the top eigenvector can change by 90° by changing one point in the data set. Relaxations such as smoothed sensitivity (Nissim et al., 2007) are difficult to compute in this setting as well. The SUb Linear Queries (SULQ) method of Blum et al. (2005) adds noise to the second moment matrix and then runs PCA on the noisy matrix. As our experiments show, the noise level required by SULQ may severely impact the quality of approximation, making it impractical for data sets of moderate size.

The goal of this paper is to characterize the problem of differentially private PCA. We assume that the algorithm is given n data points and a target dimension k and must produce a k-dimensional subspace that approximates that produced by the standard PCA problem. We propose a new algorithm, PPCA, which is an instance of the exponential mechanism of McSherry and Talwar (2007). Unlike SULQ, PPCA explicitly takes into account the quality of approximation—it outputs a k-dimensional subspace which is biased towards subspaces close to the output of PCA. In our case, the method corresponds to sampling from the matrix Bingham distribution. We implement PPCA using a Markov Chain Monte Carlo (MCMC) procedure due to Hoff (2009); simulations show that the subspace produced by PPCA captures more of the variance of A than SULQ. When the MCMC procedure converges, the algorithm provides differential privacy.

In order to understand the performance gap, we prove sample complexity bounds for the case of $k = 1$ for SULQ and PPCA, as well as a general lower bound on the sample complexity for any differentially private algorithm. We show that the sample complexity scales as $\Omega(d^{3/2}\sqrt{\log d})$ for SULQ and as $O(d)$ for PPCA. Furthermore, we show that any differentially private algorithm requires $\Omega(d)$ samples. Therefore PPCA is nearly optimal in terms of sample complexity as a function of data dimension. These theoretical results suggest that our experiments demonstrate the limit of how well ε_p-differentially private algorithms can perform, and our experiments show that this gap should persist for general k. The result seems pessimistic for many applications, because

the sample complexity depends on the extrinsic dimension d rather than the intrinsic dimension k. However, we believe this is a consequence of the fact that we make minimal assumptions on the data; our results imply that, absent additional limitations on the data set, the sample complexity differentially private PCA must grow linearly with the data dimension.

There are several interesting open questions suggested by this work. One set of issues is computational. Differentially privacy is a mathematical definition, but algorithms must be implemented using finite precision machines. Privacy and computation interact in many places, including pseudorandomness, numerical stability, optimization, and in the MCMC procedure we use to implement PPCA; investigating the impact of approximate sampling is an avenue for future work. A second set of issues is theoretical—while the privacy guarantees of PPCA hold for all k, our theoretical analysis of sample complexity applies only to $k = 1$ in which the distance and angles between vectors are related. An interesting direction is to develop theoretical bounds for general k; challenges here are providing the right notion of approximation of PCA, and extending the theory using packings of Grassmann or Stiefel manifolds. Finally, in this work we assume k is given to the algorithm, but in many applications k is chosen after looking at the data. Under differential privacy, the selection of k itself must be done in a differentially private manner.

1.1 Related Work

Differential privacy was first proposed by Dwork et al. (2006b). There has been an extensive literature following this work in the computer science theory, machine learning, and databases communities. A survey of some of the theoretical work can be found in the survey by Dwork and Smith (2009). Differential privacy has been shown to have strong *semantic* guarantees (Dwork et al., 2006b; Kasiviswanathan and Smith, 2008) and is resistant to many attacks (Ganta et al., 2008) that succeed against alternative definitions of privacy. In particular, so-called syntactic definitions of privacy (Sweeney, 2002; Machanavajjhala et al., 2006; Li et al., 2010) may be susceptible to attacks based on side-information about individuals in the database.

There are several general approaches to constructing differentially private approximations to some desired algorithm or computation. Input perturbation (Blum et al., 2005) adds noise to the data prior to performing the desired computation, whereas output perturbation (Dwork et al., 2006b) adds noise to the output of the desired computation. The exponential mechanism (McSherry and Talwar, 2007) can be used to perform differentially private selection based on a score function that measures the quality of different outputs. Objective perturbation (Chaudhuri et al., 2011) adds noise to the objective function for algorithms which are convex optimizations. These approaches and related ideas such as Nissim et al. (2007) and Dwork and Lei (2009) have been used to approximate a variety of statistical, machine learning, and data mining tasks under differential privacy (Barak et al., 2007; Wasserman and Zhou, 2010; Smith, 2011; McSherry and Mironov, 2009; Williams and McSherry, 2010; Chaudhuri et al., 2011; Rubinstein et al., 2012; Nissim et al., 2007; Blum et al., 2008; McSherry and Talwar, 2007; Friedman and Schuster, 2010; Hardt and Roth, 2012).

This paper deals with the problem of differentially private approximations to PCA. Prior to our work, the only proposed method for PCA was the Sub-Linear Queries (SULQ) method of Blum et al. (2005). This approach adds noise to the second moment matrix of the data before calculating the singular value decomposition. By contrast, our algorithm, PPCA, uses the exponential mechanism (McSherry and Talwar, 2007) to choose a k-dimensional subspace biased toward those which capture more of "energy" of the matrix. Subsequent to our work, Kapralov and Talwar (2013)

have proposed a dynamic programming algorithm for differentially private low rank matrix approximation which involves sampling from a distribution induced by the exponential mechanism. The running time of their algorithm is $O(d^6)$, where d is the data dimension, and it is unclear how this may affect its implementation. Hardt and Roth (Hardt and Roth, 2012, 2013) have studied low-rank matrix approximation under additional incoherence assumptions on the data. In particular, Hardt and Roth (2012) consider the problem of differentially-private low-rank matrix reconstruction for applications to sparse matrices; provided certain coherence conditions hold, they provide an algorithm for constructing a rank $2k$ approximation B to a matrix A such that $\|A-B\|_{\mathrm{F}}$ is $O(\|A-A_k\|)$ plus some additional terms which depend on d, k and n; here A_k is the best rank k approximation to A. Hardt and Roth (2013) show a method for guaranteeing (ε,δ)-differential privacy under an *entry-wise* neighborhood condition using the power method for calculating singular values. They, like Kapralov and Talwar (2013), also prove bounds under spectral norm perturbations, and their algorithm achieves the same error rates but with running time that is nearly linear in the number of non-zeros in the data.

In addition to these works, other researchers have examined the interplay between projections and differential privacy. Zhou et al. (2009) analyze a differentially private data release scheme where a random linear transformation is applied to data to preserve differential privacy, and then measures how much this transformation affects the utility of a PCA of the data. One example of a random linear transformation is random projection, popularized by the Johnson-Lindenstrauss (JL) transform. Blocki et al. (2012) show that the JL transform of the data preserves differential privacy provided the minimum singular value of the data matrix is large. Kenthapadi et al. (2013) study the problem of estimating the distance between data points with differential privacy using a random projection of the data points.

There has been significant work on other notions of privacy based on manipulating entries within the database (Sweeney, 2002; Machanavajjhala et al., 2006; Li et al., 2010), for example by reducing the resolution of certain features to create ambiguities. For more details on these and other alternative notions of privacy see Fung et al. (2010) for a survey with more references. An alternative line of privacy-preserving data-mining work (Zhan and Matwin, 2007) is in the Secure Multiparty Computation setting; one work (Han et al., 2009) studies privacy-preserving singular value decomposition in this model. Finally, dimension reduction through random projection has been considered as a technique for sanitizing data prior to publication (Liu et al., 2006); our work differs from this line of work in that we offer differential privacy guarantees, and we only release the PCA subspace, not actual data.

2. Preliminaries

The data given to our algorithm is a set of n vectors $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$ where each x_i corresponds to the private value of one individual, $x_i \in \mathbb{R}^d$, and $\|x_i\| \leq 1$ for all i. Let $X = [x_1, \ldots, x_n]$ be the matrix whose columns are the data vectors $\{x_i\}$. Let $A = \frac{1}{n} XX^T$ denote the $d \times d$ second moment matrix of the data. The matrix A is positive semidefinite, and has Frobenius norm $\|A\|_F$ at most 1.

The problem of dimensionality reduction is to find a "good" low-rank approximation to A. A popular solution is to compute a rank-k matrix $\hat{A}$ which minimizes the norm $\|A-\hat{A}\|_{\mathrm{F}}$, where k is much lower than the data dimension d. The Schmidt approximation theorem (Stewart, 1993) shows that the minimizer is given by the singular value decomposition, also known as the PCA algorithm in some areas of computer science.

Definition 1 *Suppose A is a positive semidefinite matrix whose first k eigenvalues are distinct. Let the eigenvalues of A be $\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_d(A) \geq 0$ and let Λ be a diagonal matrix with $\Lambda_{ii} = \lambda_i(A)$. The matrix A decomposes as*

$$A = V \Lambda V^T, \tag{1}$$

where V is an orthonormal matrix of eigenvectors. The top-k PCA subspace of A is the matrix

$$V_k(A) = [v_1 \; v_2 \; \cdots \; v_k], \tag{2}$$

where v_i is the i-th column of V in (1)*. The k-th eigengap is $\Delta_k = \lambda_k - \lambda_{k+1}$.*

Given the top-k subspace and the eigenvalue matrix Λ, we can form an approximation $A^{(k)} = V_k(A)\Lambda_k V_k(A)^T$ to A, where Λ_k contains the k largest eigenvalues in Λ. In the special case $k = 1$ we have $A^{(1)} = \lambda_1(A) v_1 v_1^T$, where v_1 is the eigenvector corresponding to $\lambda_1(A)$. We refer to v_1 as the *top eigenvector* of the data, and $\Delta = \Delta_1$ is the eigengap. For a $d \times k$ matrix $\hat{V}$ with orthonormal columns, the quality of $\hat{V}$ in approximating $V_k(A)$ can be measured by

$$q_{\mathrm{F}}(\hat{V}) = \operatorname{tr}\left(\hat{V}^T A \hat{V}\right). \tag{3}$$

The $\hat{V}$ which maximizes $q(\hat{V})$ has columns equal to $\{v_i : i \in [k]\}$, corresponding to the top-k eigenvectors of A.

Our theoretical results on the utility of our PCA approximation apply to the special case $k = 1$. We prove results about the inner product between the output vector $\hat{v}_1$ and the true top eigenvector v_1:

$$q_{\mathrm{A}}(\hat{v}_1) = |\langle \hat{v}_1, v_1 \rangle|. \tag{4}$$

The utility in (4) is related to (3). If we write $\hat{v}_1$ in the basis spanned by $\{v_i\}$, then

$$q_{\mathrm{F}}(\hat{v}_1) = \lambda_1 q_{\mathrm{A}}(\hat{v}_1)^2 + \sum_{i=2}^{d} \lambda_i \langle \hat{v}_1, v_i \rangle^2.$$

Our proof techniques use the geometric properties of $q_{\mathrm{A}}(\cdot)$.

Definition 2 *A randomized algorithm $\mathcal{A}(\cdot)$ is an (ρ, η)-close approximation to the top eigenvector if for all data sets $\mathcal{D}$ of n points we have*

$$\mathbb{P}\left(q_{\mathrm{A}}(\mathcal{A}(\mathcal{D})) \geq \rho\right) \geq 1 - \eta,$$

where the probability is taken over $\mathcal{A}(\cdot)$.

We study approximations to $\mathcal{A}$ to PCA that preserve the privacy of the underlying data. The notion of privacy that we use is differential privacy, which quantifies the privacy guaranteed by a randomized algorithm $\mathcal{A}$ applied to a data set $\mathcal{D}$.

Definition 3 *An algorithm $\mathcal{A}(\mathcal{B})$ taking values in a set $\mathcal{T}$ provides ε_p-differential privacy if*

$$\sup_{\mathcal{S}} \sup_{\mathcal{D},\mathcal{D}'} \frac{\mu(\mathcal{S} \mid \mathcal{B} = \mathcal{D})}{\mu(\mathcal{S} \mid \mathcal{B} = \mathcal{D}')} \leq e^{\varepsilon_p},$$

where the first supremum is over all measurable $\mathcal{S} \subseteq \mathcal{T}$, the second is over all data sets $\mathcal{D}$ and $\mathcal{D}'$ differing in a single entry, and $\mu(\cdot|\mathcal{B})$ is the conditional distribution (measure) on $\mathcal{T}$ induced by the output $\mathcal{A}(\mathcal{B})$ given a data set $\mathcal{B}$. The ratio is interpreted to be 1 whenever the numerator and denominator are both 0.

Definition 4 *An algorithm $\mathcal{A}(\mathcal{B})$ taking values in a set $\mathcal{T}$ provides (ε_p, δ)-differential privacy if*

$$\mathbb{P}(\mathcal{A}(\mathcal{D}) \in \mathcal{S}) \leq e^{\varepsilon_p} \mathbb{P}\left(\mathcal{A}(\mathcal{D}') \in \mathcal{S}\right) + \delta,$$

for all measurable $\mathcal{S} \subseteq \mathcal{T}$ and all data sets $\mathcal{D}$ and $\mathcal{D}'$ differing in a single entry.

Here ε_p and δ are privacy parameters, where low ε_p and δ ensure more privacy (Dwork et al., 2006b; Wasserman and Zhou, 2010; Dwork et al., 2006a). The second privacy guarantee is weaker; the parameter δ bounds the probability of failure, and is typically chosen to be quite small. In our experiments we chose small but constant δ—Ganta et al. (2008) suggest $\delta < \frac{1}{n^2}$ is more appropriate.

In this paper we are interested in proving results on the sample complexity of differentially private algorithms that approximate PCA. That is, for a given ε_p and ρ, how large must the number of individuals n in the data set be such that the algorithm is both ε_p-differentially private and a (ρ, η)-close approximation to PCA? It is well known that as the number of individuals n grows, it is easier to guarantee the same level of privacy with relatively less noise or perturbation, and therefore the utility of the approximation also improves. Our results characterize how the privacy ε_p and utility ρ scale with n and the tradeoff between them for fixed n. We show that the sample complexity depends on the eigengap Δ.

3. Algorithms and Results

In this section we describe differentially private techniques for approximating (2). The first is a modified version of the Sub-Linear Queries (SULQ) method (Blum et al., 2005). Our new algorithm for differentially-private PCA, PPCA, is an instantiation of the exponential mechanism due to McSherry and Talwar (2007). Both procedures are differentially private approximations to the top-k subspace: SULQ guarantees (ε_p, δ)-differential privacy and PPCA guarantees ε_p-differential privacy.

3.1 Input Perturbation

The only differentially-private approximation to PCA prior to this work is the SULQ method (Blum et al., 2005). The SULQ method perturbs each entry of the empirical second moment matrix A to ensure differential privacy and releases the top-k eigenvectors of this perturbed matrix. More specifically, SULQ recommends adding a matrix N of i.i.d. Gaussian noise of variance $\frac{8d^2 \log^2(d/\delta)}{n^2 \varepsilon_p^2}$ and applies the PCA algorithm to $A + N$. This guarantees a weaker privacy definition known as (ε_p, δ)-differential privacy. One problem with this approach is that with probability 1 the matrix $A+N$ is not symmetric, so the largest eigenvalue may not be real and the entries of the corresponding

eigenvector may be complex. Thus the SULQ algorithm, as written, is not a good candidate for approximating PCA.

It is easy to modify SULQ to produce a an eigenvector with real entries that guarantees (ε_p, δ) differential privacy. In Algorithm 1, instead of adding an asymmetric Gaussian matrix, we add a symmetric matrix with i.i.d. Gaussian entries N. That is, for $1 \leq i \leq j \leq d$, the variable N_{ij} is an independent Gaussian random variable with variance β^2. Note that this matrix is symmetric but not necessarily positive semidefinite, so some eigenvalues may be negative but the eigenvectors are all real. A derivation for the noise variance in (5) of Algorithm 1 is given in Theorem 5. An alternative is to add Laplace noise of an appropriate variance to each entry—this would guarantee ε_p-differential privacy.

Algorithm 1: Algorithm MOD-SULQ (input pertubation)

inputs: $d \times n$ data matrix X, privacy parameter ε_p, parameter δ

outputs: $d \times k$ matrix $\hat{V}_k = [\hat{v}_1\ \hat{v}_2\ \cdots\ \hat{v}_k]$ with orthonormal columns

1 Set $A = \frac{1}{n} XX^T$.;

2 Set

$$\beta = \frac{d+1}{n\varepsilon_p}\sqrt{2\log\left(\frac{d^2+d}{\delta 2\sqrt{2\pi}}\right)} + \frac{1}{n\sqrt{\varepsilon_p}}. \tag{5}$$

Generate a $d \times d$ symmetric random matrix N whose entries are i.i.d. drawn from $\mathcal{N}\left(0, \beta^2\right)$. ;

3 Compute $\hat{V}_k = V_k(A+N)$ according to (2). ;

3.2 Exponential Mechanism

Our new method, PPCA, randomly samples a k-dimensional subspace from a distribution that ensures differential privacy and is biased towards high utility. The distribution from which our released subspace is sampled is known in the statistics literature as the matrix Bingham distribution (Chikuse, 2003), which we denote by $\mathsf{BMF}_k(B)$. The algorithm and its privacy properties apply to general $k < d$ but our theoretical results on the utility focus on the special case $k = 1$. The matrix Bingham distribution takes values on the set of all k-dimensional subspaces of $\mathbb{R}^d$ and has a density equal to

$$f(V) = \frac{1}{{}_1F_1\left(\frac{1}{2}k, \frac{1}{2}d, B\right)} \exp(\mathrm{tr}(V^T BV)), \tag{6}$$

where V is a $d \times k$ matrix whose columns are orthonormal and ${}_1F_1\left(\frac{1}{2}k, \frac{1}{2}d, B\right)$ is a confluent hypergeometric function (Chikuse, 2003, p.33).

By combining results on the exponential mechanism along with properties of PCA algorithm, we can show that this procedure is differentially private. In many cases, sampling from the distribution specified by the exponential mechanism may be expensive computationally, especially for continuous-valued outputs. We implement PPCA using a recently-proposed Gibbs sampler due to Hoff (2009). Gibbs sampling is a popular Markov Chain Monte Carlo (MCMC) technique in which samples are generated according to a Markov chain whose stationary distribution is the density in

Algorithm 2: Algorithm PPCA (exponential mechanism)

inputs: $d \times n$ data matrix X, privacy parameter ε_p, dimension k

outputs: $d \times k$ matrix $\hat{V}_k = [\hat{v}_1 \ \hat{v}_2 \ \cdots \ \hat{v}_k]$ with orthonormal columns

1 Set $A = \frac{1}{n} X X^T$;

2 Sample $\hat{V}_k = \mathsf{BMF}\left(n\frac{\varepsilon_p}{2} A\right)$;

(6). Assessing the "burn-in time" and other factors for this procedure is an interesting question in its own right; further details are in Section 6.2.

3.3 Other Approaches

There are other general algorithmic strategies for guaranteeing differential privacy. The sensitivity method (Dwork et al., 2006b) adds noise proportional to the maximum change that can be induced by changing a single point in the data set. Consider a data set $\mathcal{D}$ with $m+1$ copies of a unit vector u and m copies of a unit vector u' with $u \perp u'$ and let $\mathcal{D}'$ have m copies of u and $m+1$ copies of u'. Then $v_1(\mathcal{D}) = u$ but $v_1(\mathcal{D}') = u'$, so $\|v_1(\mathcal{D}) - v_1(\mathcal{D}')\| = \sqrt{2}$. Thus the global sensitivity does not scale with the number of data points, so as n increases the variance of the noise required by the sensitivity method will not decrease. An alternative to global sensitivity is smooth sensitivity (Nissim et al., 2007). Except for special cases, such as the sample median, smooth sensitivity is difficult to compute for general functions. A third method for computing private, approximate solutions to high-dimensional optimization problems is objective perturbation (Chaudhuri et al., 2011); to apply this method, we require the optimization problems to have certain properties (namely, strong convexity and bounded norms of gradients), which do not apply to PCA.

3.4 Main Results

Our theoretical results are sample complexity bounds for PPCA and MOD-SULQ as well as a general lower bound on the sample complexity for any ε_p-differentially private algorithm. These results show that the PPCA is nearly optimal in terms of the scaling of the sample complexity with respect to the data dimension d, privacy parameter ε_p, and eigengap Δ. We further show that MOD-SULQ requires more samples as a function of d, despite having a slightly weaker privacy guarantee. Proofs are presented in Sections 4 and 5.

Even though both algorithms can output the top-k PCA subspace for general $k \leq d$, we prove results for the case $k = 1$. Finding the scaling behavior of the sample complexity with k is an interesting open problem that we leave for future work; challenges here are finding the right notion of approximation of the PCA, and extending the theory using packings of Grassman or Stiefel manifolds.

Theorem 5 *For the* β *in* (5) *Algorithm MOD-SULQ is* (ε_p, δ) *differentially private.*

Theorem 6 *Algorithm PPCA is* ε_p*-differentially private.*

The fact that these two algorithms are differentially private follows from some simple calculations. Our first sample complexity result provides an upper bound on the number of samples required by PPCA to guarantee a certain level of privacy and accuracy. The sample complexity of

PPCA grows linearly with the dimension d, inversely with ε_p, and inversely with the correlation gap $(1-\rho)$ and eigenvalue gap Δ. These sample complexity results hold for $k=1$.

Theorem 7 (Sample complexity of PPCA) *If*

$$n > \frac{d}{\varepsilon_p \Delta(1-\rho)}\left(4\frac{\log(1/\eta)}{d} + 2\log\frac{8\lambda_1}{(1-\rho^2)\Delta}\right),$$

then the top PCA direction v_1 and the output of PPCA $\hat{v}_1$ with privacy parameter ε_p satisfy

$$\Pr(|\langle v_1, \hat{v}_1\rangle| > \rho) \geq 1-\eta.$$

That is, PPCA is a (ρ,η)-close approximation to PCA.

Our second result shows a lower bound on the number of samples required by *any* ε_p-differentially-private algorithm to guarantee a certain level of accuracy for a large class of data sets, and uses proof techniques in Chaudhuri and Hsu (2011, 2012).

Theorem 8 (Sample complexity lower bound) *Fix $d \geq 3$, ε_p, $\Delta \leq \frac{1}{2}$ and let $1-\phi = \exp\left(-2\cdot\frac{\ln 8+\ln(1+\exp(d))}{d-2}\right)$. For any $\rho \geq 1-\frac{1-\phi}{16}$, no ε_p-differentially private algorithm $\mathcal{A}$ can approximate PCA with expected utility greater than ρ on all databases with n points in dimension d having eigenvalue gap Δ, where*

$$n < \frac{d}{\varepsilon_p \Delta}\max\left\{1, \sqrt{\frac{1-\phi}{80(1-\rho)}}\right\}.$$

Theorem 7 shows that if n scales like $\frac{d}{\varepsilon_p\Delta(1-\rho)}\log\frac{1}{1-\rho^2}$ then PPCA produces an approximation $\hat{v}_1$ that has correlation ρ with v_1, whereas Theorem 8 shows that n must scale like $\frac{d}{\varepsilon_p\Delta\sqrt{(1-\rho)}}$ for any ε_p-differentially private algorithm. In terms of scaling with d, ε_p and Δ, the upper and lower bounds match, and they also match up to square-root factors with respect to the correlation. By contrast, the following lower bound on the number of samples required by MOD-SULQ to ensure a certain level of accuracy shows that MOD-SULQ has a less favorable scaling with dimension.

Theorem 9 (Sample complexity lower bound for MOD-SULQ) *There are constants c and c' such that if*

$$n < c\frac{d^{3/2}\sqrt{\log(d/\delta)}}{\varepsilon_p}(1-c'(1-\rho)),$$

then there is a data set of size n in dimension d such that the top PCA direction v and the output $\hat{v}$ of MOD-SULQ satisfy $\mathbb{E}[|\langle \hat{v}_1, v_1\rangle|] \leq \rho$.

Notice that the dependence on n grows as $d^{3/2}$ in SULQ as opposed to d in PPCA. Dimensionality reduction via PCA is often used in applications where the data points occupy a low dimensional space but are presented in high dimensions. These bounds suggest that PPCA is better suited to such applications than MOD-SULQ.

4. Analysis of PPCA

In this section we provide theoretical guarantees on the performance of PPCA. The proof of Theorem 6 follows from the results on the exponential mechanism (McSherry and Talwar, 2007). To find the sample complexity of PPCA we bound the density of the Bingham distribution, leading to a sample complexity for $k = 1$ that depends on the gap $\lambda_1 - \lambda_2$ between the top two eigenvalues. We also prove a general lower bound on the sample complexity that holds for any ε_p-differentially private algorithm. The lower bound matches our upper bound up to log factors, showing that PPCA is nearly optimal in terms of the scaling with dimension, privacy ε_p, and utility $q_{\mathrm{A}}(\cdot)$.

4.1 Privacy Guarantee

We first give a proof of Theorem 6.

Proof Let X be a data matrix whose i-th column is x_i and $A = \frac{1}{n}XX^T$. The PP-PCA algorithm is the exponential mechanism of McSherry and Talwar (2007) applied to the score function $n \cdot v^T A v$. Consider $X' = [x_1\ x_2\ \cdots\ x_{n-1}\ x_n']$ differing from X in a single column and let $A' = \frac{1}{n}X'X'^T$. We have

$$\begin{aligned}\max_{v\in\mathbb{S}^{d-1}} \left|n\cdot v^T A' v - n\cdot v^T A v\right| &\leq \left|v^T(x_n' x_n'^T - x_n x_n^T)v\right| \\ &\leq \left|\left\|v^T x_n'\right\|^2 - \left\|v^T x_n\right\|^2\right| \\ &\leq 1.\end{aligned}$$

The last step follows because $\|x_i\| \leq 1$ for all i. The result now follows immediately from McSherry and Talwar (2007, Theorem 6). ■

4.2 Upper Bound on Utility

The results on the exponential mechanism bound the gap between the value of the function $q_{\mathrm{F}}(\hat{v}_1) = n\cdot\hat{v}_1^T A\hat{v}_1$ evaluated at the output $\hat{v}_1$ of the mechanism and the optimal value $q(v_1) = n\cdot\lambda_1$. We derive a bound on the correlation $q_{\mathrm{A}}(\hat{v}_1) = |\langle\hat{v}_1, v_1\rangle|$ via geometric arguments.

Lemma 10 (Lemmas 2.2 and 2.3 of Ball (1997)) *Let μ be the uniform measure on the unit sphere $\mathbb{S}^{d-1}$. For any $x \in \mathbb{S}^{d-1}$ and $0 \leq c < 1$ the following bounds hold:*

$$\frac{1}{2}\exp\left(-\frac{d-1}{2}\log\frac{2}{1-c}\right) \leq \mu\left(\left\{v\in\mathbb{S}^{d-1} : \langle v,x\rangle \geq c\right\}\right) \leq \exp\left(-dc^2/2\right).$$

We are now ready to provide a proof of Theorem 7.

Proof Fix a privacy level ε_p, target correlation ρ, and probability η. Let X be the data matrix and $B = (\varepsilon_p/2)XX^T$ and

$$\mathcal{U}_\rho = \{u : |\langle u, v_1\rangle| \geq \rho\}.$$

be the union of the two spherical caps centered at $\pm v_1$. Let $\overline{\mathcal{U}}_\rho$ denote the complement of $\mathcal{U}_\rho$ in $\mathbb{S}^{d-1}$.

An output vector $\hat{v}_1$ is "good" if it is in $\mathcal{U}_\rho$. We first give some bounds on the score function $q_{\mathrm{F}}(u)$ on the boundary between $\mathcal{U}_\rho$ and $\overline{\mathcal{U}}_\rho$, where $\langle u, v_1\rangle = \pm\rho$. On this boundary, the function

$q_{\mathrm{F}}(u)$ is maximized when u is a linear combination of v_1 and v_2, the top two eigenvectors of A. It minimized when u is a linear combination of v_1 and v_d. Therefore

$$q_{\mathrm{F}}(u) \leq \frac{n\varepsilon_p}{2}(\rho^2\lambda_1 + (1-\rho^2)\lambda_2) \qquad u \in \overline{\mathcal{U}}_\rho, \tag{7}$$

$$q_{\mathrm{F}}(u) \geq \frac{n\varepsilon_p}{2}(\rho^2\lambda_1 + (1-\rho^2)\lambda_d) \qquad u \in \mathcal{U}_\rho. \tag{8}$$

Let $\mu(\cdot)$ denote the uniform measure on the unit sphere. Then fixing an $0 \leq b < 1$, using (7), (8), and the fact that $\lambda_d \geq 0$,

$$\begin{aligned}
\mathbb{P}\left(\overline{\mathcal{U}}_\rho\right) &\leq \frac{\mathbb{P}\left(\overline{\mathcal{U}}_\rho\right)}{\mathbb{P}\left(\mathcal{U}_\sigma\right)} \\
&= \frac{\frac{1}{{}_1F_1\left(\frac{1}{2}k,\frac{1}{2}m,B\right)}\int_{\overline{\mathcal{U}}_\rho}\exp\left(u^TBu\right)d\mu}{\frac{1}{{}_1F_1\left(\frac{1}{2}k,\frac{1}{2}m,B\right)}\int_{\mathcal{U}_\sigma}\exp\left(u^TBu\right)d\mu} \\
&\leq \frac{\exp\left(n(\varepsilon_p/2)\left(\rho^2\lambda_1+(1-\rho^2)\lambda_2\right)\right)\cdot\mu\left(\overline{\mathcal{U}}_\rho\right)}{\exp\left(n(\varepsilon_p/2)\left(\sigma^2\lambda_1+(1-\sigma^2)\lambda_d\right)\right)\cdot\mu\left(\mathcal{U}_\sigma\right)} \\
&\leq \exp\left(-\frac{n\varepsilon_p}{2}\left(\sigma^2\lambda_1-(\rho^2\lambda_1+(1-\rho^2)\lambda_2)\right)\right)\cdot\frac{\mu\left(\overline{\mathcal{U}}_\rho\right)}{\mu\left(\mathcal{U}_\sigma\right)}.
\end{aligned} \tag{9}$$

Applying the lower bound from Lemma 10 to the denominator of (9) and the upper bound $\mu\left(\overline{\mathcal{U}}_\rho\right) \leq 1$ yields

$$\mathbb{P}\left(\overline{\mathcal{U}}_\rho\right) \leq \exp\left(-\frac{n\varepsilon_p}{2}\left(\sigma^2\lambda_1-(\rho^2\lambda_1+(1-\rho^2)\lambda_2)\right)\right)\cdot\exp\left(\frac{d-1}{2}\log\frac{2}{1-\sigma}\right). \tag{10}$$

We must choose a $\sigma^2 > \rho^2$ to make the upper bound smaller than 1. More precisely,

$$\begin{aligned}
\sigma^2 &> \rho^2 + (1-\rho^2)\frac{\lambda_2}{\lambda_1}, \\
1-\sigma^2 &< (1-\rho^2)\left(1-\frac{\lambda_2}{\lambda_1}\right).
\end{aligned}$$

For simplicity, choose

$$1-\sigma^2 = \frac{1}{2}(1-\rho^2)\left(1-\frac{\lambda_2}{\lambda_1}\right).$$

So that

$$\begin{aligned}
\sigma^2\lambda_1-(\rho^2\lambda_1+(1-\rho^2)\lambda_2) &= (1-\rho^2)\lambda_1-(1-\sigma^2)\lambda_1-(1-\rho^2)\lambda_2 \\
&= (1-\rho^2)\left(\lambda_1-\frac{1}{2}(\lambda_1-\lambda_2)-\lambda_2\right) \\
&= \frac{1}{2}(1-\rho^2)(\lambda_1-\lambda_2)
\end{aligned}$$

and

$$\log\frac{2}{1-\sigma} < \log\frac{4}{1-\sigma^2} = \log\frac{8\lambda_1}{(1-\rho^2)(\lambda_1-\lambda_2)}.$$

Setting the right hand side of (10) less than η yields

$$\frac{n\varepsilon_p}{4}(1-\rho^2)(\lambda_1-\lambda_2) > \log\frac{1}{\eta} + \frac{d-1}{2}\log\frac{8\lambda_1}{(1-\rho^2)(\lambda_1-\lambda_2)}.$$

Because $1-\rho < 1-\rho^2$, if we choose

$$n > \frac{d}{\varepsilon_p(1-\rho)(\lambda_1-\lambda_2)}\left(4\frac{\log(1/\eta)}{d} + 2\log\frac{8\lambda_1}{(1-\rho^2)(\lambda_1-\lambda_2)}\right),$$

then the output of PPCA will produce a $\hat{v}_1$ such that

$$\mathbb{P}(|\langle\hat{v}_1, v_1\rangle| < \rho) < \eta.$$

■

4.3 Lower Bound on Utility

We now turn to a general lower bound on the sample complexity for any differentially private approximation to PCA. We construct K databases which differ in a small number of points whose top eigenvectors are not too far from each other. For such a collection, Lemma 12 shows that for any differentially private mechanism, the average correlation over the collection cannot be too large. That is, any ε_p-differentially private mechanism cannot have high utility on all K data sets. The remainder of the argument is to construct these K data sets.

The proof uses some simple eigenvalue and eigenvector computations. A matrix of positive entries

$$A = \begin{pmatrix} a & b \\ b & c \end{pmatrix} \tag{11}$$

has characteristic polynomial

$$\det(A-\lambda I) = \lambda^2 - (a+c)\lambda + (ac-b^2)$$

and eigenvalues

$$\lambda = \frac{1}{2}(a+c) \pm \frac{1}{2}\sqrt{(a+c)^2 - 4(ac-b^2)} = \frac{1}{2}(a+c) \pm \frac{1}{2}\sqrt{(a-c)^2+4b^2}.$$

The eigenvectors are in the directions $(b, -(a-\lambda))^T$.

We will also need the following Lemma, which is proved in the Appendix.

Lemma 11 (Simple packing set) *For $\phi \in [(2\pi d)^{-1/2}, 1)$, there exists a set of*

$$K = \frac{1}{8}\exp\left((d-1)\log\frac{1}{\sqrt{1-\phi^2}}\right) \tag{12}$$

vectors $\mathcal{C}$ in $\mathbb{S}^{d-1}$ such that for any pair $\mu, \nu \in \mathcal{C}$, the inner product between them is upper bounded by ϕ:

$$|\langle \mu, \nu \rangle| \leq \phi.$$

The following Lemma gives a lower bound on the expected utility averaged over a set of databases which differ in a "small" number of elements.

Lemma 12 *Let $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$ be K databases which differ in the value of at most $\frac{\ln(K-1)}{\varepsilon_p}$ points, and let $u_1, \ldots, u_K$ be the top eigenvectors of $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$. If $\mathcal{A}$ is any ε_p-differentially private algorithm, then,*

$$\sum_{i=1}^{K} \mathbb{E}_{\mathcal{A}}\left[|\langle \mathcal{A}(\mathcal{D}_i), u_i \rangle|\right] \leq K\left(1 - \frac{1}{16}(1 - \max\left|\langle u_i, u_j \rangle\right|)\right).$$

Proof Let

$$t = \min_{i \neq j}(\left\|u_i - u_j\right\|, \left\|u_i + u_j\right\|),$$

and $\mathcal{G}_i$ be the "double cap" around $\pm u_i$ of radius $t/2$:

$$\mathcal{G}_i = \{u : \|u - u_i\| < t/2\} \cup \{u : \|u + u_i\| < t/2\}.$$

We claim that

$$\sum_{i=1}^{K} \mathbb{P}_{\mathcal{A}}(\mathcal{A}(\mathcal{D}_i) \notin G_i) \geq \frac{1}{2}(K-1). \tag{13}$$

The proof is by contradiction. Suppose the claim is false. Because all of the caps $\mathcal{G}_i$ are disjoint, and applying the definition of differential privacy,

$$\begin{aligned}
\frac{1}{2}(K-1) &> \sum_{i=1}^{K} \mathbb{P}_{\mathcal{A}}(\mathcal{A}(\mathcal{D}_i) \notin G_i) \\
&\geq \sum_{i=1}^{K} \sum_{i' \neq i} \mathbb{P}_{\mathcal{A}}(\mathcal{A}(\mathcal{D}_i) \in G_{i'}) \\
&\geq \sum_{i=1}^{K} \sum_{i' \neq i} e^{-\varepsilon_p \cdot \ln(K-1)/\varepsilon_p} \mathbb{P}_{\mathcal{A}}(\mathcal{A}(\mathcal{D}_{i'}) \in G_{i'}) \\
&\geq (K-1) \cdot \frac{1}{K-1} \cdot \sum_{i=1}^{K} \mathbb{P}_{\mathcal{A}}(\mathcal{A}(\mathcal{D}_i) \in G_i) \\
&\geq K - \frac{1}{2}(K-1),
\end{aligned}$$

which is a contradiction, so (13) holds. Therefore by the Markov inequality

$$\sum_{i=1}^{K} \mathbb{E}_{\mathcal{A}} \left[\min(\|\mathcal{A}(\mathcal{D}_i) - u_i\|^2, \|\mathcal{A}(\mathcal{D}_i) + u_i\|^2) \right] \geq \sum_{i=1}^{K} \mathbb{P}(\mathcal{A}(\mathcal{D}_i) \notin G_i) \cdot \frac{t^2}{4}$$
$$\geq \frac{1}{8}(K-1)t^2.$$

Rewriting the norms in terms of inner products shows

$$2K - 2\sum_{i=1}^{K} \mathbb{E}_{\mathcal{A}} \left[|\langle \mathcal{A}(\mathcal{D}_i), u_i \rangle| \right] \geq \frac{1}{8}(K-1)\left(2 - 2\max \left|\langle u_i, u_j \rangle\right|\right),$$

so

$$\sum_{i=1}^{K} \mathbb{E}_{\mathcal{A}} \left[|\langle \mathcal{A}(\mathcal{D}_i), u_i \rangle| \right] \leq K \left(1 - \frac{1}{8}\frac{K-1}{K}(1 - \max \left|\langle u_i, u_j \rangle\right|) \right)$$
$$\leq K \left(1 - \frac{1}{16}(1 - \max \left|\langle u_i, u_j \rangle\right|) \right).$$

■

We can now prove Theorem 8.

Proof From Lemma 12, given a set of K databases differing in $\frac{\ln(K-1)}{\varepsilon_p}$ points with top eigenvectors $\{u_i : i = 1, 2, \ldots, K\}$, for at least one database i,

$$\mathbb{E}_{\mathcal{A}} \left[|\langle \mathcal{A}(\mathcal{D}_i), u_i \rangle| \right] \leq 1 - \frac{1}{16}\left(1 - \max \left|\langle u_i, u_j \rangle\right|\right)$$

for any ε_p-differentially private algorithm. Setting the left side equal to some target ρ,

$$1 - \rho \geq \frac{1}{16}\left(1 - \max \left|\langle u_i, u_j \rangle\right|\right). \tag{14}$$

So our goal is construct these data bases such that the inner product between their eigenvectors is small.

Let $y = e_d$, the d-th coordinate vector, and let $\phi \in ((2\pi d)^{-1/2}, 1)$. Lemma 11 shows that there exists a packing $\mathcal{W} = \{w_1, w_2, \ldots, w_K\}$ of the sphere $\mathbb{S}^{d-2}$ spanned by the first $d-1$ elementary vectors $\{e_1, e_2, \ldots, e_{d-1}\}$ such that $\max_{i \neq j} |\langle w_i, w_j \rangle| \leq \phi$, where

$$K = \frac{1}{8}(1 - \phi)^{-(d-2)/2}.$$

Choose ϕ such that $\ln(K-1) = d$. This means

$$1 - \phi = \exp\left(-2 \cdot \frac{\ln 8 + \ln(1 + \exp(d))}{d-2} \right).$$

The right side is minimized for $d = 3$ but this leads to a weak lower bound $1 - \phi > 3.5 \times 10^{-5}$. By contrast, for $d = 100$, the bound is $1 - \phi > 0.12$. In all cases, $1 - \phi$ is at least a constant value.

We construct a database with n points for each w_i. Let $\beta = \frac{d}{n\varepsilon_p}$. For now, we assume that $\beta \leq \Delta \leq \frac{1}{2}$. The other case, when $\beta \geq \Delta$ will be considered later. Because $\beta \leq \Delta$, we have

$$n > \frac{d}{\varepsilon_p \Delta}.$$

The construction uses a parameter $0 \leq m \leq 1$ that will be set as a function of the eigenvalue gap Δ. We will derive conditions on n based on the requirements on d, ε_p, ρ, and Δ. For $i = 1, 2, \ldots, K$ let the data set $\mathcal{D}_i$ contain

- $n(1-\beta)$ copies of $\sqrt{m}y$
- $n\beta$ copies of $z_i = \frac{1}{\sqrt{2}}y + \frac{1}{\sqrt{2}}w_i$.

Thus data sets $\mathcal{D}_i$ and $\mathcal{D}_j$ differ in the values of $n\beta = \frac{\ln(K-1)}{n\varepsilon_p}$ individuals. The second moment matrix A_i of $\mathcal{D}_i$ is

$$A_i = ((1-\beta)m + \frac{1}{2}\beta)yy^T + \frac{1}{2}\beta(w_i^T y + yw_i^T) + \frac{1}{2}\beta w_i w_i^T.$$

By choosing an basis containing y and w_i, we can write this as

$$A_i = \begin{bmatrix} (1-\beta)m + \frac{1}{2}\beta & \frac{1}{2}\beta & \mathbf{0} \\ \frac{1}{2}\beta & \frac{1}{2}\beta & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

This is in the form (11), with $a = (1-\beta)m + \frac{1}{2}\beta$, $b = \frac{1}{2}\beta$, and $c = \frac{1}{2}\beta$.

The matrix A_i has two nonzero eigenvalues given by

$$\begin{aligned} \lambda &= \frac{1}{2}(a+c) + \frac{1}{2}\sqrt{(a-c)^2 + 4b^2}, \\ \lambda' &= \frac{1}{2}(a+c) - \frac{1}{2}\sqrt{(a-c)^2 + 4b^2}, \end{aligned} \tag{15}$$

The gap Δ between the top two eigenvalues is:

$$\Delta = \sqrt{(a-c)^2 + 4b^2} = \sqrt{m^2(1-\beta)^2 + \beta^2}.$$

We can thus set m in the construction to ensure an eigengap of Δ:

$$m = \frac{\sqrt{(\Delta^2 - \beta^2)}}{1-\beta}. \tag{16}$$

The top eigenvector of A_i is given by

$$u_i = \frac{b}{\sqrt{b^2 + (a-\lambda)^2}}y + \frac{(a-\lambda)}{\sqrt{b^2 + (a-\lambda)^2}}w_i.$$

where λ is given by (15). Therefore

$$\begin{aligned}\max_{i\neq j}\left|\langle u_i,u_j\rangle\right| &\leq \frac{b^2}{b^2+(a-\lambda)^2}+\frac{(a-\lambda)^2}{b^2+(a-\lambda)^2}\max_{i\neq j}\left|\langle w_i,w_j\rangle\right| \\ &\leq 1-\frac{(a-\lambda)^2}{b^2+(a-\lambda)^2}(1-\phi). \end{aligned} \tag{17}$$

To obtain an upper bound on $\max_{i\neq j}\left|\langle u_i,u_j\rangle\right|$ we must lower bound $\frac{(a-\lambda)^2}{b^2+(a-\lambda)^2}$.

Since $x/(\nu+x)$ is monotonically increasing in x when $\nu>0$, we will find a lower bound on $(a-\lambda)$. Observe that from (15),

$$\lambda-a=\frac{b^2}{\lambda-c}.$$

So to lower bound $\lambda-a$ we need to upper bound $\lambda-c$. We have

$$\lambda-c=\frac{1}{2}(a-c)+\frac{1}{2}\Delta=\frac{1}{2}\left((1-\beta)m+\Delta\right).$$

Because $b=\beta/2$,

$$(\lambda-a)^2>\left(\frac{\beta^2}{2((1-\beta)m+\Delta)}\right)^2=\frac{\beta^4}{4((1-\beta)m+\Delta)^2}.$$

Now,

$$\begin{aligned}\frac{(a-\lambda)^2}{b^2+(a-\lambda)^2} &> \frac{\beta^4}{\beta^2((1-\beta)m+\Delta)^2+\beta^4} \\ &= \frac{\beta^2}{\beta^2+((1-\beta)m+\Delta)^2} \\ &> \frac{\beta^2}{5\Delta^2}, \end{aligned} \tag{18}$$

where the last step follows by plugging in m from (16) and using the fact that $\beta\leq\Delta$. Putting it all together, we have from (14), (17), and (18), and using the fact that ϕ is such that $\ln(K-1)=d$ and $\beta=\frac{d}{n\varepsilon_p}$,

$$\begin{aligned}1-\rho &\geq \frac{1}{16}\cdot\frac{(a-\lambda)^2}{b^2+(a-\lambda)^2}(1-\phi) \\ &> \frac{1-\phi}{80}\frac{\beta^2}{\Delta^2} \\ &= \frac{1-\phi}{80}\cdot\frac{d^2}{n^2\varepsilon_p^2\Delta^2}, \end{aligned}$$

which implies

$$n>\frac{d}{\varepsilon_p\Delta}\sqrt{\frac{1-\phi}{80(1-\rho)}}.$$

Thus for $\beta \leq \Delta \leq 1/2$, any ε_p-differentially private algorithm needs $\Omega\left(\frac{d}{\varepsilon_p \Delta \sqrt{1-\rho}}\right)$ points to get expected inner product ρ on all data sets with eigengap Δ.

We now consider the case where $\beta > \Delta$. We choose a slightly different construction here. The i-th database now consists of $n(1-\beta)$ copies of the 0 vector, and $n\beta$ copies of $\frac{\Delta}{\beta} w_i$. Thus, every pair of databases differ in the values of $n\beta = \frac{\ln(K-1)}{\varepsilon_p}$ people, and the eigenvalue gap between the top two eigenvectors is $\beta \cdot \frac{\Delta}{\beta} = \Delta$.

As the top eigenvector of the i-th database is $u_i = w_i$,

$$\max_{i \neq j} |\langle u_i, u_j \rangle| = \max_{i \neq j} |\langle w_i, w_j \rangle| \leq \phi.$$

Combining this with (14), we obtain

$$1 - \rho \geq \frac{1}{16}(1-\phi),$$

which provides the additional condition in the Theorem. ■

5. Analysis of MOD-SULQ

In this section we provide theoretical guarantees on the performance of the MOD-SULQ algorithm. Theorem 5 shows that MOD-SULQ is (ε_p, δ)-differentially private. Theorem 15 provides a lower bound on the distance between the vector released by MOD-SULQ and the true top eigenvector in terms of the privacy parameters ε_p and δ and the number of points n in the data set. This implicitly gives a lower bound on the sample complexity of MOD-SULQ. We provide some graphical illustration of this tradeoff.

The following upper bound will be useful for future calculations : for two unit vectors x and y,

$$\sum_{1 \leq i \leq j \leq d} (x_i x_j - y_i y_j)^2 \leq 2. \tag{19}$$

Note that this upper bound is achievable by setting x and y to be orthogonal elementary vectors.

5.1 Privacy Guarantee

We first justify the choice of β^2 in the MOD-SULQ algorithm by proving Theorem 5.

Proof Let B and $\hat{B}$ be two independent symmetric random matrices where $\{B_{ij} : 1 \leq i \leq j \leq d\}$ and $\{\hat{B}_{ij} : 1 \leq i \leq j \leq d\}$ are each sets of i.i.d. Gaussian random variables with mean 0 and variance β^2. Consider two data sets $\mathcal{D} = \{x_i : i = 1, 2, \ldots, n\}$ and $\hat{\mathcal{D}} = \mathcal{D}_1 \cup \{\hat{x}_n\} \setminus \{x_n\}$ and let A and $\hat{A}$ denote their second moment matrices. Let $G = A + B$ and $\hat{G} = \hat{A} + \hat{B}$. We first calculate the log ratio of the densities of G and $\hat{G}$ at a point H:

$$\begin{aligned}
\log \frac{f_G(H)}{f_{\hat{G}}(H)} &= \sum_{1 \leq i \leq j \leq d} \left(-\frac{1}{2\beta^2}(H_{ij} - A_{ij})^2 + \frac{1}{2\beta^2}(H_{ij} - \hat{A}_{ij})^2 \right) \\
&= \frac{1}{2\beta^2} \sum_{1 \leq i \leq j \leq d} \left(\frac{2}{n}(H_{ij} - A_{ij})(x_{n,i}x_{n,j} - \hat{x}_{n,i}\hat{x}_{n,j}) + \frac{1}{n^2}(\hat{x}_{n,i}\hat{x}_{n,j} - x_{n,i}x_{n,j})^2 \right).
\end{aligned}$$

From (19) the last term is upper bounded by $2/n^2$. To upper bound the first term,

$$\begin{aligned}\sum_{1\le i\le j\le d} |\hat{x}_{n,i}\hat{x}_{n,j} - x_{n,i}x_{n,j}| &\le 2 \max_{a:\|a\|\le 1} \sum_{1\le i\le j\le d} a_i a_j \\ &\le 2\cdot\frac{1}{2}(d^2+d)\cdot\frac{1}{d} \\ &= d+1.\end{aligned}$$

Note that this bound is not too loose—by taking $\hat{x} = d^{-1/2}\mathbf{1}$ and $x = (1,0,\ldots,0)^T$, this term is still linear in d.

Then for any measurable set $\mathcal{S}$ of matrices,

$$\mathbb{P}(G\in\mathcal{S}) \le \exp\left(\frac{1}{2\beta^2}\left(\frac{2}{n}(d+1)\gamma + \frac{3}{n^2}\right)\right)\mathbb{P}\left(\hat{G}\in\mathcal{S}\right) + \mathbb{P}(B_{ij} > \gamma \text{ for all } i,j). \tag{20}$$

To handle the last term, use a union bound over the $(d^2+d)/2$ variables $\{B_{ij}\}$ together with the tail bound, which holds for $\gamma > \beta$:

$$\mathbb{P}(B_{ij} > \gamma) \le \frac{1}{\sqrt{2\pi}} e^{-\gamma^2/2\beta^2}.$$

Thus setting $\mathbb{P}(B_{ij} > \gamma \text{ for some } i,j) = \delta$ yields the condition

$$\delta = \frac{d^2+d}{2\sqrt{2\pi}} e^{-\gamma^2/2\beta^2}.$$

Rearranging to solve for γ gives

$$\gamma = \max\left(\beta, \beta\sqrt{2\log\left(\frac{d^2+d}{\delta 2\sqrt{2\pi}}\right)}\right) = \beta\sqrt{2\log\left(\frac{d^2+d}{\delta 2\sqrt{2\pi}}\right)}$$

for $d > 1$ and $\delta < 3/\sqrt{2\pi e}$. This then gives an expression for ε_p to make (20) imply (ε_p, δ) differential privacy:

$$\begin{aligned}\varepsilon_p &= \frac{1}{2\beta^2}\left(\frac{2}{n}(d+1)\gamma + \frac{2}{n^2}\right) \\ &= \frac{1}{2\beta^2}\left(\frac{2}{n}(d+1)\beta\sqrt{2\log\left(\frac{d^2+d}{\delta 2\sqrt{2\pi}}\right)} + \frac{2}{n^2}\right).\end{aligned}$$

Solving for β using the quadratic formula yields the particularly messy expression in (5):

$$\begin{aligned}\beta &= \frac{d+1}{2n\varepsilon_p}\sqrt{2\log\left(\frac{d^2+d}{\delta 2\sqrt{2\pi}}\right)} + \frac{1}{2n\varepsilon_p}\left(2(d+1)^2\log\left(\frac{d^2+d}{\delta 2\sqrt{2\pi}}\right) + 4\varepsilon_p\right)^{1/2} \\ &\le \frac{d+1}{n\varepsilon_p}\sqrt{2\log\left(\frac{d^2+d}{\delta 2\sqrt{2\pi}}\right)} + \frac{1}{\sqrt{\varepsilon_p}n}.\end{aligned}$$

■

5.2 Proof of Theorem 9

In this section we provide theoretical guarantees on the performance of the MOD-SULQ algorithm. Theorem 5 shows that MOD-SULQ is (ε_p, δ)-differentially private. Theorem 15 provides a lower bound on the distance between the vector released by MOD-SULQ and the true top eigenvector in terms of the privacy parameters ε_p and δ and the number of points n in the data set. This implicitly gives a lower bound on the sample complexity of MOD-SULQ. We provide some graphical illustration of this tradeoff. The main tool in our lower bound is a generalization by Yu (1997) of an information-theoretic inequality due to Fano.

Theorem 13 (Fano's inequality (Yu, 1997)) *Let $\mathcal{R}$ be a set and Θ be a parameter space with a pseudo-metric $d(\cdot)$. Let $\mathcal{F}$ be a set of r densities $\{f_1, \ldots, f_r\}$ on $\mathcal{R}$ corresponding to parameter values $\{\theta_1, \ldots, \theta_r\}$ in Θ. Let X have distribution $f \in \mathcal{F}$ with corresponding parameter θ and let $\hat{\theta}(X)$ be an estimate of θ. If, for all i and j*

$$d(\theta_i, \theta_j) \geq \tau$$

and

$$\mathbf{KL}(f_i \| f_j) \leq \gamma,$$

then

$$\max_j \mathbb{E}_j[d(\hat{\theta}, \theta_j)] \geq \frac{\tau}{2}\left(1 - \frac{\gamma + \log 2}{\log r}\right),$$

where $\mathbb{E}_j[\cdot]$ denotes the expectation with respect to distribution f_j.

To use this inequality, we will construct a set of densities on the set of covariance matrices corresponding distribution of the random matrix in the MOD-SULQ algorithm under different inputs. These inputs will be chosen using a set of unit vectors which are a packing on the surface of the unit sphere.

Lemma 14 *Let Σ be a positive definite matrix and let f denote the density $\mathcal{N}(a, \Sigma)$ and g denote the density $\mathcal{N}(b, \Sigma)$. Then $\mathbf{KL}(f \| g) = \frac{1}{2}(a-b)^T \Sigma^{-1}(a-b)$.*

Proof This is a simple calculation:

$$\begin{aligned}
\mathbf{KL}(f \| g) &= \mathbb{E}_{x \sim f}\left[-\frac{1}{2}(x-a)^T\Sigma^{-1}(x-a) + \frac{1}{2}(x-b)\Sigma^{-1}(x-b)\right] \\
&= \frac{1}{2}\left(a^T\Sigma^{-1}a - a^T\Sigma^{-1}b - b^T\Sigma^{-1}a + b^T\Sigma^{-1}b\right) \\
&= \frac{1}{2}(a-b)^T\Sigma^{-1}(a-b).
\end{aligned}$$

■

The next theorem is a lower bound on the expected distance between the vector output by MOD-SULQ and the true top eigenvector. In order to get this lower bound, we construct a class of data sets and use Theorem 13 to derive a bound on the minimax error over the class.

Theorem 15 (Utility bound for MOD-SULQ) *Let d, n, and $\varepsilon_p > 0$ be given and let β be given by Algorithm 1 so that the output of MOD-SULQ is (ε_p, δ)-differentially private for all data sets in $\mathbb{R}^d$ with n elements. Then there exists a data set with n elements such that if $\hat{v}_1$ denotes the output of MOD-SULQ and v_1 is the top eigenvector of the empirical covariance matrix of the data set, the expected correlation $\langle \hat{v}_1, v_1 \rangle$ is upper bounded:*

$$\mathbb{E}\left[|\langle \hat{v}_1, v_1 \rangle|\right] \leq \min_{\phi \in \Phi} \left(1 - \frac{(1-\phi)}{4} \left(1 - \frac{1/\beta^2 + \log 2}{(d-1)\log \frac{1}{\sqrt{1-\phi^2}} - \log(8)} \right)^2 \right), \tag{21}$$

where

$$\Phi \in \left[\max \left\{ \frac{1}{\sqrt{2\pi d}}, \sqrt{1 - \exp\left(-\frac{2\log(8d)}{d-1} \right)}, \sqrt{1 - \exp\left(-\frac{2/\beta^2 + \log(256)}{d-1} \right)} \right\}, 1 \right). \tag{22}$$

Proof For $\phi \in [(2\pi d)^{-1/2}, 1)$, Lemma 11 shows there exists a set of K unit vectors $\mathcal{C}$ such that for $\mu, \nu \in \mathcal{C}$, the inner product between them satisfies $|\langle \mu, \nu \rangle| < \phi$, where K is given by (12). Note that for small ϕ this setting of K is loose, but any orthonormal basis provides d unit vectors which are orthogonal, setting $K = d$ and solving for ϕ yields

$$\left(1 - \exp\left(-\frac{2\log(8d)}{d-1} \right) \right)^{1/2}.$$

Setting the lower bound on ϕ to the maximum of these two yields the set of ϕ and K which we will consider in (22).

For any unit vector μ, let

$$A(\mu) = \mu\mu^T + N,$$

where N is a $d \times d$ symmetric random matrix such that $\{N_{ij} : 1 \leq i \leq j \leq d\}$ are i.i.d. $\mathcal{N}(0, \beta^2)$, where β^2 is the noise variance used in the MOD-SULQ algorithm. Due to symmetry, the matrix $A(\mu)$ can be thought of as a jointly Gaussian random vector on the $d(d+1)/2$ variables $\{A_{ij}(\mu) : 1 \leq i \leq j \leq d\}$. The mean of this vector is

$$\bar{\mu} = \left(\mu_1^2, \mu_2^2, \ldots, \mu_d^2, \mu_1\mu_2, \mu_1\mu_3, \ldots, \mu_{d-1}\mu_d \right)^T,$$

and the covariance is $\beta^2 I_{d(d+1)/2}$. Let f_μ denote the density of this vector.

For $\mu, \nu \in \mathcal{C}$, the divergence between f_μ and f_ν can be calculated using Lemma 14:

$$\begin{aligned} \mathbf{KL}\left(f_\mu \| f_\nu\right) &= \frac{1}{2}(\bar{\mu} - \bar{\nu})^T \Sigma^{-1} (\bar{\mu} - \bar{\nu}) \\ &= \frac{1}{2\beta^2} \|\bar{\mu} - \bar{\nu}\|^2 \\ &\leq \frac{1}{\beta^2}. \end{aligned} \tag{23}$$

The last line follows from the fact that the vectors in $\mathcal{C}$ are unit norm.

For any two vectors $\mu, \nu \in \mathcal{C}$, lower bound the Euclidean distance between them using the upper bound on the inner product:

$$\|\mu - \nu\| \geq \sqrt{2(1-\phi)}. \tag{24}$$

Let $\Theta = \mathbb{S}^{d-1}$ with the Euclidean norm and $\mathcal{R}$ be the set of distributions $\{A(\mu) : \mu \in \Theta\}$. From (24) and (23), the set $\mathcal{C}$ satisfies the conditions of Theorem 13 with $\mathcal{F} = \{f_\mu : \mu \in \mathcal{C}\}$, $r = K$, $\tau = \sqrt{2(1-\phi)}$, and $\gamma = \frac{1}{\beta^2}$. The conclusion of the Theorem shows that for MOD-SULQ,

$$\max_{\mu \in \mathcal{C}} \mathbb{E}_{f_\mu}[\|\hat{v} - \mu\|] \geq \frac{\sqrt{2(1-\phi)}}{2}\left(1 - \frac{1/\beta^2 + \log 2}{\log K}\right). \tag{25}$$

This lower bound is vacuous when the term inside the parenthesis is negative, which imposes further conditions on ϕ. Setting $\log K = 1/\beta^2 + \log 2$, we can solve to find another lower bound on ϕ:

$$\phi \geq \sqrt{1 - \exp\left(-\frac{2/\beta^2 + \log(256)}{d-1}\right)}.$$

This yields the third term in (22). Note that for larger n this term will dominate the others.

Using Jensen's inequality on the the left side of (25):

$$\max_{\mu \in \mathcal{C}} \mathbb{E}_{f_\mu}[2(1 - |\langle \hat{v}, \mu \rangle|)] \geq \frac{(1-\phi)}{2}\left(1 - \frac{1/\beta^2 + \log 2}{\log K}\right)^2.$$

So there exists a $\mu \in \mathcal{C}$ such that

$$\mathbb{E}_{f_\mu}[|\langle \hat{v}, \mu \rangle|] \leq 1 - \frac{(1-\phi)}{4}\left(1 - \frac{1/\beta^2 + \log 2}{\log K}\right)^2. \tag{26}$$

Consider the data set consisting of n copies of μ. The corresponding covariance matrix is $\mu\mu^T$ with top eigenvector $v_1 = \mu$. The output of the algorithm MOD-SULQ applied to this data set is an estimator of μ and hence satisfies (26). Minimizing over ϕ gives the desired bound. ■

The minimization over ϕ in (21) does not lead to analytically pretty results, so we plotted the results in Figure 1 in order to get a sense of the bounds. Figure 1 shows the lower bound on the expected correlation $\mathbb{E}[|\langle \hat{v}_1, v_1 \rangle|]$ as a function of the number of data points (given on a logarithmic scale). Each panel shows a different dimension, from $d = 50$ to $d = 1000$, and plots are given for different values of ε_p ranging from 0.01 to 2. In all experiments we set $\delta = 0.01$. In high dimension, the lower bound shows that the expected performance of MOD-SULQ is poor when there are a small number of data points. This limitation may be particularly acute when the data lies in a very low dimensional subspace but is presented in very high dimension. In such "sparse" settings, perturbing the input as in MOD-SULQ is not a good approach. However, in lower dimensions and data-rich regimes, the performance may be more favorable.

A little calculation yields the sample complexity bound in Theorem 9

Proof Suppose $\mathbb{E}[|\langle \hat{v}_1, v_1 \rangle|] = \rho$. Then a little algebra shows

$$2\sqrt{1-\rho} \geq \min_{\phi \in \Phi} \sqrt{1-\phi}\left(1 - \frac{1/\beta^2 + \log 2}{(d-1)\log \frac{1}{\sqrt{1-\phi^2}} - \log(8)}\right).$$

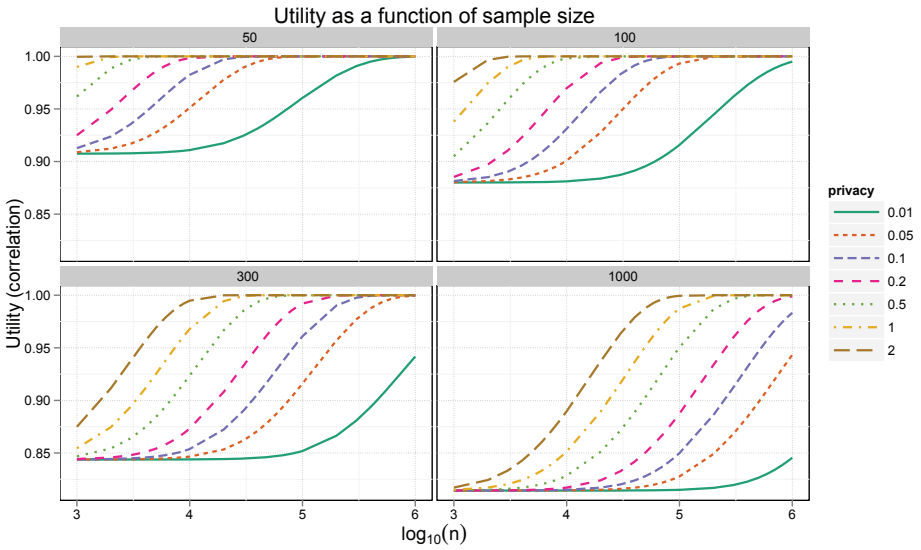


Figure 1: Upper bound from Theorem 15 on the expected correlation between the true top eigenvector and the $\hat{v}_1$ produced by MOD-SULQ. The horizontal axis is $\log_{10}(n)$ and the vertical axis shows the lower bound in (21). The four panels correspond to different values of the dimension d, from 50 to 1000. Each panel contains plots of the bound for different values of ε_p.

Setting ϕ such that $(d-1)\log\frac{1}{\sqrt{1-\phi^2}} - \log(8) = 2(1/\beta^2 + \log 2)$ we have

$$4\sqrt{1-\rho} \geq \sqrt{1-\phi}.$$

Since we are concerned with the scaling behavior for large d and n, this implies

$$\log\frac{1}{\sqrt{1-\phi^2}} = \Theta\left(\frac{1}{\beta^2 d}\right),$$

so

$$\begin{aligned}\phi &= \sqrt{1-\exp\left(-\Theta\left(\frac{1}{\beta^2 d}\right)\right)} \\ &= \Theta\left(\sqrt{\frac{1}{\beta^2 d}}\right).\end{aligned}$$

From Algorithm 1, to get for some constant c_1, we have the following lower bound on β:

$$\beta^2 > c_1 \frac{d^2}{n^2\varepsilon_p^2}\log(d/\delta).$$

Substituting, we get for some constants c_2 and c_3 that

$$(1 - c_2(1-\rho)) \leq c_3 \frac{n^2 \varepsilon_p^2}{d^3 \log(d/\delta)}.$$

Now solving for n shows

$$n \geq c \frac{d^{3/2}\sqrt{\log(d/\delta)}}{\varepsilon_p} \left(1 - c'(1-\rho)\right).$$

■

6. Experiments

We next turn to validating our theoretical results on real data. We implemented MOD-SULQ and PPCA in order to test our theoretical bounds. Implementing PPCA involved using a Gibbs sampling procedure (Hoff, 2009). A crucial parameter in MCMC procedures is the burn-in time, which is how long the chain must be run for it to reach its stationary distribution. Theoretically, chains reach their stationary distribution only in the limit; however, in practice MCMC users must sample after some finite time. In order to use this procedure appropriately, we determined a burn-in time using our data sets. The interaction of MCMC procedures and differential privacy is a rich area for future research.

6.1 Data and Preprocessing

We report on the performance of our algorithm on some real data sets. We chose four data sets from four different domains—`kddcup99` (Bache and Lichman, 2013), which includes features of 494,021 network connections, `census` (Bache and Lichman, 2013), a demographic data set on 199,523 individuals, `localization` (Kaluža et al., 2010), a medical data set with 164,860 instances of sensor readings on individuals engaged in different activities, and `insurance` (van der Putten and van Someren, 2000), a data set on product usage and demographics of 9,822 individuals.

These data sets contain a mix of continuous and categorical features. We preprocessed each data set by converting a feature with q discrete values to a vector in $\{0,1\}^q$; after preprocessing, the data sets `kddcup99`, `census`, `localization` and `insurance` have dimensions 116, 513, 44 and 150 respectively. We also normalized each row so that each entry has maximum value 1, and normalize each column such that the maximum (Euclidean) column norm is 1. We choose $k = 4$ for `kddcup`, $k = 8$ for `census`, $k = 10$ for `localization` and $k = 11$ for `insurance`; in each case, the utility $q_{\mathrm{F}}(V_k)$ of the top-k PCA subspace of the data matrix accounts for at least 80% of $\|A\|_{\mathrm{F}}$. Thus, all four data sets, although fairly high dimensional, have good low-dimensional representations. The properties of each data set are summarized in Table 1.

6.2 Implementation of Gibbs Sampling

The theoretical analysis of PPCA uses properties of the Bingham distribution $\mathsf{BMF}_k(\cdot)$ given in (6). To implement this algorithm for experiments we use a Gibbs sampler due to Hoff (2009). The Gibbs sampling scheme induces a Markov Chain, the stationary distribution of which is the density in (6).

Data Set	#instances	#dimensions	k	$q_{\mathrm{F}}(V_k)$	$q_{\mathrm{F}}(V_k)/\|A\|_{\mathrm{F}}$
kddcup	494,021	116	4	0.6587	0.96
census	199,523	513	8	0.7321	0.81
localization	164,860	44	10	0.5672	0.81
insurance	9,822	150	11	0.5118	0.81

Table 1: Parameters of each data set. The second column is the number of dimensions after preprocessing. k is the dimensionality of the PCA, the third column contains $q_{\mathrm{F}}(V_k)$, where V_k is the top-k PCA subspace, and the fifth column is the normalized utility $q_{\mathrm{F}}(V_k)/\|A\|_{\mathrm{F}}$.

Gibbs sampling and other MCMC procedures are widely used in statistics, scientific modeling, and machine learning to estimate properties of complex distributions (Brooks, 1998).

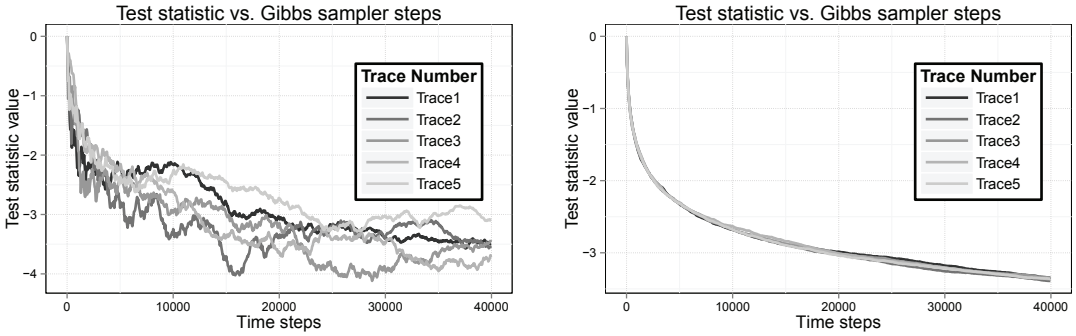
Finding the speed of convergence of MCMC methods is still an open area of research. There has been much theoretical work on estimating convegence times (Jones and Hobart, 2004; Douc et al., 2004; Jones and Hobart, 2001; Roberts, 1999; Roberts and Sahu, 2001; Roberts, 1999; Roberts and Sahu, 2001; Rosenthal, 1995; Kolassa, 1999, 2000), but unfortunately, most theoretical guarantees are available only in special cases and are often too weak for practical use. In lieu of theoretical guarantees, users of MCMC methods empirically estimate the *burn-in time*, or the number of iterations after which the chain is sufficiently close to its stationary distribution. Statisticians employ a range of diagnostic methods and statistical tests to empirically determine if the Markov chain is close to stationarity (Cowles and Carlin, 1996; Brooks and Roberts, 1998; Brooks and Gelman, 1998; El Adlouni et al., 2006). These tests do not provide a sufficient guarantee of stationarity, and there is no "best test" to use. In practice, the convergence of derived statistics is used to estimate an appropriate the burn-in time. In the case of the Bingham distribution, Hoff (2009) performs qualitative measures of convergence. Developing a better characterization of the convergence of this Gibbs sampler is an important question for future work.

Because the MCMC procedure of Hoff (2009) does not come with convergence-time guarantees, for our experiments we had to choose an appropriate burn-in time. The "ideal" execution of PPCA provides ε_p-differential privacy, but because our implementation only approximates sampling from the Bingham distribution, we cannot guarantee that this implementation provides the privacy guarantee. As noted by Mironov (2012), even current implementations of floating-point arithmetic may suffer from privacy problems, so there is still significant work to do between theory and implementation. For this paper we tried to find a burn-in time that was sufficiently long so that we could be confident that the empirical performance of PPCA was not affected by the initial conditions of the sampler.

In order to choose an appropriate burn-in time, we examined the *time series trace* of the Markov Chain. We ran l copies, or traces, of the chain, starting from l different initial locations drawn uniformly from the set of all $d \times k$ matrices with orthonormal columns. Let $X^i(t)$ be the output of the i-th copy at iteration t, and let U be the top-k PCA subspace of the data. We used the following statistic as a function of iteration T:

$$F_k^i(T) = \frac{1}{\sqrt{k}} \left\| \frac{1}{T} \sum_{t=1}^{T} X^i(t) \right\|_F ,$$

where $||\cdot||_F$ is the Frobenius norm. The matrix Bingham distribution has mean 0, and hence with increasing T, the statistic $F_k^i(T)$ should converge to 0.



(a) `kddcup` ($k = 4$) (b) `insurance` ($k = 11$)

Figure 2: Plots of $\log F_k^i(T)$ for five different traces (values of i) on two different data sets. Figure 2(a) shows $\log F_k^i(T)$ for for $k = 4$ as a function of iteration T for $40,000$ steps of the Gibbs sampler on the `kddcup` data set. Figure 2(b) shows the same for the `insurance` data set.

Figure 2 illustrates the behavior of the Gibbs sampler. The plots show the value of $\log F_k^i(T)$ as a function of the Markov chain iteration for 5 different restarts of the MCMC procedure for two data sets, `kddcup` and `insurance`. The initial starting points were chosen uniformly from the set of all $d \times k$ matrices with orthonormal columns. The plots show that $F_k^i(T)$ decreases rapidly after a few thousand iterations, and is less than 0.01 after $T = 20{,}000$ in both cases. $\log F_k^i(T)$ also appears to have a larger variance for `kddcup` than for `insurance`; this is explained by the fact that the `kddcup` data set has a much larger number of samples, which makes its stationary distribution farther from the initial distribution of the sampler. Based on these and other simulations, we observed that the Gibbs sampler converges to $F_k(t) < 0.01$ at $t = 20{,}000$ when run on data with a few hundred dimensions and with k between 5 and 10; we thus chose to run the Gibbs sampler for $T = 20{,}000$ timesteps for all the data sets.

Our simulations indicate that the chains converge fairly rapidly, particularly when $\|A - A_k\|_{\mathrm{F}}$ is small so that A_k is a good approximation to A. Convergence is slower for larger n when the initial state is chosen from the uniform distribution over all $k \times d$ matrices with orthonormal columns; this is explained by the fact that for larger n, the stationary distribution is farther in variation distance from the starting distribution, which results in a longer convergence time.

6.3 Scaling with Data Set Size

We ran three algorithms on these data sets : standard (non-private) PCA, MOD-SULQ, and PPCA. As a sanity check, we also tried a uniformly generated random projection—since this projection is data-independent we would expect it to have low utility. We measured the utility $q_{\mathrm{F}}(U)$, where U is the k-dimensional subspace output by the algorithm; $q_{\mathrm{F}}(U)$ is maximized when U is the top-k PCA subspace, and thus this reflects how close the output subspace is to the true PCA subspace in terms

of representing the data. Although our theoretical results hold for $q_{\mathrm{A}}(\cdot)$, the "energy" $q_{\mathrm{F}}(\cdot)$ is more relevant in practice for larger k.

To investigate how well these different algorithms performed on real data, for each data set we subsampled data sets of different sizes n uniformly and ran the algorithms on the subsets. We chose $\varepsilon_p = 0.1$ for this experiment, and for MOD-SULQ we used $\delta = 0.01$. We averaged over 5 such subsets and over several instances of the randomized algorithms (10 restarts for PPCA and 100 for MOD-SULQ and random projections). For each subset and instance we calculated the resulting utility $q_{\mathrm{F}}(\cdot)$ of the output subspace.

Figures 3(a), 3(b), 4(a), and 4(b) show $q_{\mathrm{F}}(U)$ as a function of the subsampled data set sizes. The bars indicate the standard deviation over the restarts (from subsampling the data and random sampling for privacy). The non-private algorithm achieved $q_{\mathrm{F}}(V_k)$ for nearly all subset sizes (see Table 1 for the values). These plots illustrate how additional data can improve the utility of the output for a fixed privacy level ε_p. As n increases, the dashed blue line indicating the utility of PPCA begins to approach $q_{\mathrm{F}}(V_k)$, the utility of the optimal subspace.

These experiments also show that the performance of PPCA is significantly better than that of MOD-SULQ, and MOD-SULQ produces subspaces whose utility is on par with randomly choosing a subspace. The only exception to this latter point is `localization`, We believe this is because d is much lower for this data set ($d = 44$), which shows that for low dimension and large n, MOD-SULQ may produce subspaces with reasonable utility. Furthermore, MOD-SULQ is simpler and hence runs faster than PPCA, which requires running the Gibbs sampler past the burn-in time. Our theoretical results suggest that the performance of differentially private PCA cannot be significantly improved over the performance of PPCA but since those results hold for $k = 1$ they do not immediately apply here.

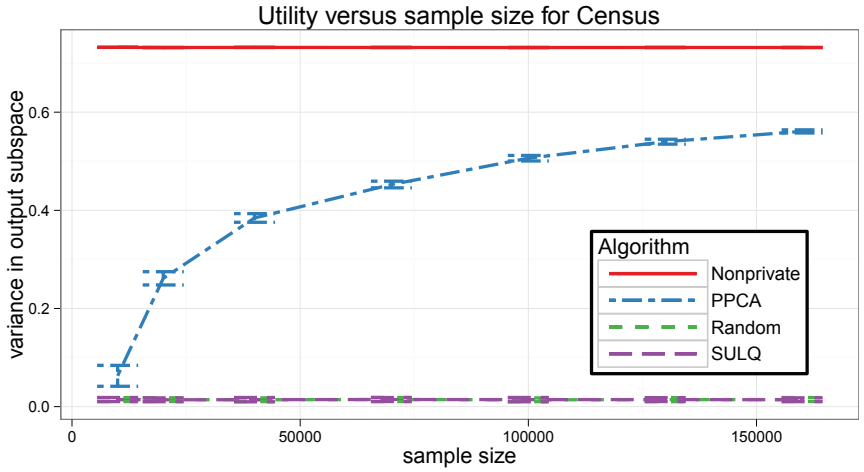
6.4 Effect of Privacy on Classification

A common use of a dimension reduction algorithm is as a precursor to classification or clustering; to evaluate the effectiveness of the different algorithms, we projected the data onto the subspace output by the algorithms, and measured the classification accuracy using the projected data. The classification results are summarized in Table 2. We chose the *normal* vs. all classification task in `kddcup99`, and the *falling* vs. all classification task in `localization`.[1] We used a linear SVM for all classification tasks, which is implemented by libSVM (Chang and Lin, 2011).

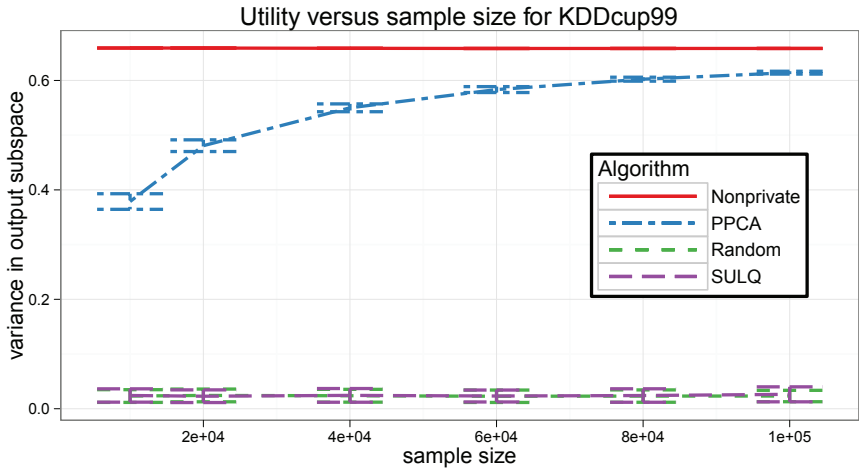
For the classification experiments, we used half of the data as a holdout set for computing a projection subspace. We projected the classification data onto the subspace computed based on the holdout set; 10% of this data was used for training and parameter-tuning, and the rest for testing. We repeated the classification process 5 times for 5 different (random) projections for each algorithm, and then ran the entire procedure over 5 random permutations of the data. Each value in the figure is thus an average over $5 \times 5 = 25$ rounds of classification.

The classification results show that our algorithm performs almost as well as non-private PCA for classification in the top-k PCA subspace, while the performance of MOD-SULQ and random projections are a little worse. The classification accuracy while using MOD-SULQ and random projections also appears to have higher variance compared to our algorithm and non-private PCA. This

1. For the other two data sets, `census` and `insurance`, the classification accuracy of linear SVM after (non-private) PCAs is as low as always predicting the majority label.



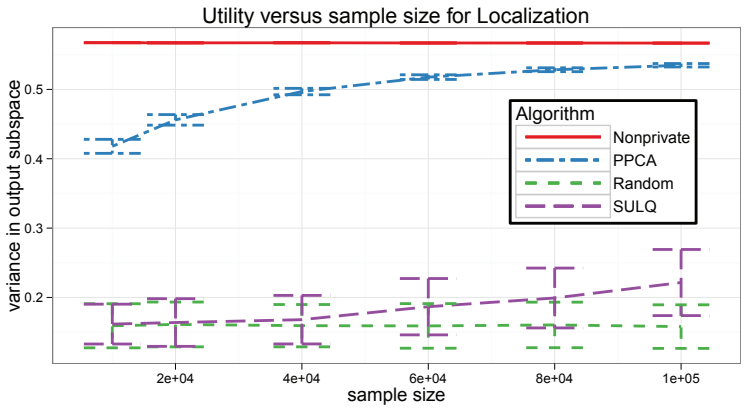
(a) `census` ($k = 8$)



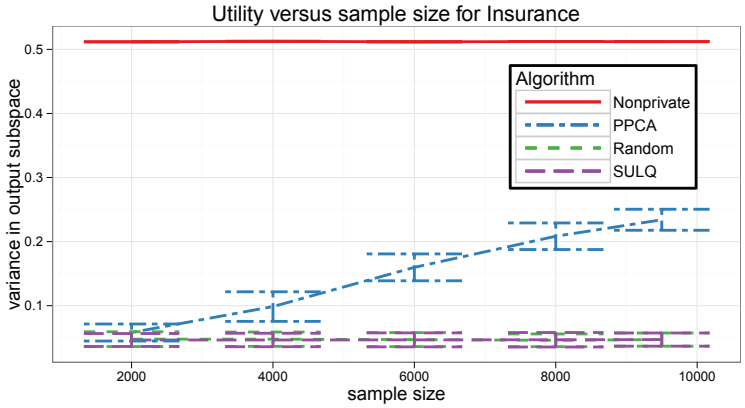
(b) `kddcup` ($k = 4$)

Figure 3: Plot of the unnormalized utility $q_{\mathrm{F}}(U)$ versus the sample size n, averaged over random subsets of the data and randomness in the algorithms. The bars are at one standard deviation about the mean. The top red line is the PCA algorithm without privacy constraints. The dashed line in blue is the utility for PPCA. The green and purple dashed lines are nearly indistinguishable and represent the utility from random projections and MOD-SULQ, respectively. In these plots $\varepsilon_p = 0.1$ and $\delta = 0.01$.

is because the projections tend to be farther from the top-k PCA subspace, making the classification error more variable.



(a) `localization` ($k = 10$)



(b) `insurance` ($k = 11$)

Figure 4: Plot of the unnormalized utility $q_{\mathrm{F}}(U)$ versus the sample size n, averaged over random subsets of the data and randomness in the algorithms. The bars are at one standard deviation about the mean. The top red line is the PCA algorithm without privacy constraints. The dashed line in blue is the utility for PPCA. The green and purple dashed lines are nearly indistinguishable for `insurance` but diverge for `localization`—they represent the utility from random projections and MOD-SULQ, respectively. In these plots $\varepsilon_p = 0.1$ and $\delta = 0.01$.

6.5 Effect of the Privacy Requirement

How to choose ε_p is important open question for many applications. We wanted to understand the impact of varying ε_p on the utility of the subspace. We did this via a synthetic data set—we generated $n = 5{,}000$ points drawn from a Gaussian distribution in $d = 10$ with mean $\mathbf{0}$ and covariance matrix with eigenvalues

$$\{0.5, 0.30, 0.04, 0.03, 0.02, 0.01, 0.004, 0.003, 0.001, 0.001\}. \tag{27}$$

	kddcup99	**localization**
Non-private PCA	98.97 ± 0.05	100 ± 0
PPCA	98.95 ± 0.05	100 ± 0
MOD-SULQ	98.18 ± 0.65	97.06 ± 2.17
Random Projections	98.23 ± 0.49	96.28 ± 2.34

Table 2: Classification accuracy in the k-dimensional subspaces for `kddcup99` ($k = 4$), and `localization` ($k = 10$) in the k-dimensional subspaces reported by the different algorithms.

In this case the space spanned by the top two eigenvectors has most of the energy, so we chose $k = 2$ and plotted the utility $q_{\mathrm{F}}(\cdot)$ for non-private PCA, MOD-SULQ with $\delta = 0.05$, and PPCA with a burn-in time of $T = 1000$. We drew 100 samples from each privacy-preserving algorithm and the plot of the average utility versus ε_p is shown in Figure 5. The privacy requirement relaxes as ε_p increases, and both MOD-SULQ and PPCA approach the utility of PCA without privacy constraints. However, for moderate ε_p PPCA still captures most of the utility, whereas the gap between MOD-SULQ and PPCA becomes quite large.

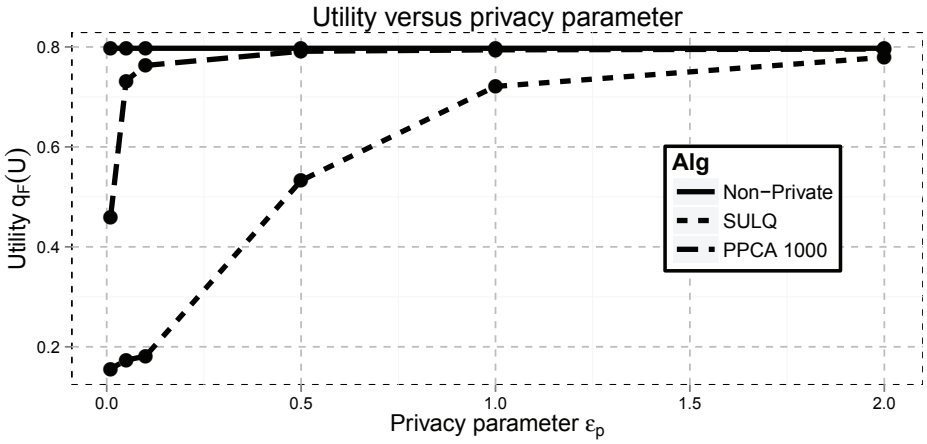


Figure 5: Plot of $q_{\mathrm{F}}(U)$ versus ε_p for a synthetic data set with $n = 5{,}000$, $d = 10$, and $k = 2$. The data has a Gaussian distribution whose covariance matrix has eigenvalues given by (27).

7. Conclusion

In this paper we investigated the theoretical and empirical performance of differentially private approximations to PCA. Empirically, we showed that MOD-SULQ and PPCA differ markedly in how well they approximate the top-k subspace of the data. The reason for this, theoretically, is that the sample complexity of MOD-SULQ scales as $d^{3/2}\sqrt{\log d}$ whereas PPCA scales as d. Because PPCA uses the exponential mechanism with $q_{\mathrm{F}}(\cdot)$ as the utility function, it is not surprising that it

performs well. However, MOD-SULQ often had a performance comparable to random projections, indicating that the real data sets had too few points for it to be effective. We furthermore showed that PPCA is nearly optimal, in that any differentially private approximation to PCA must use $\Omega(d)$ samples.

Our investigation brought up many interesting issues to consider for future work. The description of differentially private algorithms assume an ideal model of computation : real systems require additional security assumptions that have to be verified. The difference between truly random noise and pseudorandomness and the effects of finite precision can lead to a gap between the theoretical ideal and practice. Numerical optimization methods used in some privacy methods (Chaudhuri et al., 2011) can only produce approximate solutions; they may also have complex termination conditions unaccounted for in the theoretical analysis. MCMC sampling is similar : if we can guarantee that the sampler's distribution has total variation distance δ from the Bingham distribution, then sampler can guarantee (ε_p, δ) differential privacy. However, we do not yet have such analytical bounds on the convergence rate; we must determine the Gibbs sampler's convergence empirically. Accounting for these effects is an interesting avenue for future work that can bring theory and practice together.

For PCA more specifically, it would be interesting to extend the sample complexity results to general $k > 1$. For $k = 1$ the utility functions $q_{\mathrm{F}}(\cdot)$ and $q_{\mathrm{A}}(\cdot)$ are related, but for larger k it is not immediately clear what metric best captures the idea of "approximating" the top-k PCA subspace. For minimax lower bounds, it may be possible to construct a packing with respect to a general utility metric. For example, Kapralov and Talwar (2013) use properties of packings on the Grassmann manifold. Upper bounds on the sample complexity for PPCA may be possible by performing a more careful analysis of the Bingham distribution or by finding better approximations for its normalizing constant. Developing a framework for analyzing general approximations to PCA may be of interest more broadly in machine learning.

Acknowledgments

The authors would like to thank the reviewers for their detailed comments, which greatly improved the quality and readability of the manuscript, and the action editor, Gabor Lugosi, for his patience during the revision process. KC and KS would like to thank NIH for research support under U54-HL108460. The experimental results were made possible by support from the UCSD FWGrid Project, NSF Research Infrastructure Grant Number EIA-0303622. ADS was supported in part by the California Institute for Telecommunications and Information Technology (CALIT2) at UC San Diego.

Appendix A. A Packing Lemma

The proof of this lemma is relatively straightforward. The following is a slight refinement of a lemma due to Csiszár and Narayan (1988, 1991).

Lemma 16 *Let $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_N$ be arbitrary random variables and let $f_i(\mathbf{Z}_1, \ldots, \mathbf{Z}_i)$ be arbitrary with $0 \leq f_i \leq 1$, $i = 1, 2, \ldots, N$. Then the condition*

$$\mathbb{E}\left[f_i(\mathbf{Z}_1, \ldots, \mathbf{Z}_i) | \mathbf{Z}_1, \ldots, \mathbf{Z}_{i-1}\right] \leq a_i \text{ a.s.}, \qquad i = 1, 2, \ldots, N$$

implies that

$$\mathbb{P}\left(\frac{1}{N}\sum_{i=1}^{N} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i) > t\right) \leq \exp\left(-Nt(\log 2) + \sum_{i=1}^{N} a_i\right).$$

Proof First apply Markov's inequality:

$$\begin{aligned}
&\mathbb{P}\left(\frac{1}{N}\sum_{i=1}^{N} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i) > t\right) \\
&\quad = \mathbb{P}\left(2^{\sum_{i=1}^{N} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i)} > 2^{Nt}\right) \\
&\quad \leq 2^{-Nt}\mathbb{E}\left[2^{\sum_{i=1}^{N} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i)}\right] \\
&\quad \leq 2^{-Nt}\mathbb{E}\left[2^{\sum_{i=1}^{N-1} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i)}\mathbb{E}\left[2^{f_N(\mathbf{Z}_1,\ldots,\mathbf{Z}_N)}|\mathbf{Z}_1,\ldots,\mathbf{Z}_{N-1}\right]\right].
\end{aligned}$$

Now note that for $b \in [0,1]$ we have $2^b \leq 1+b \leq e^b$, so

$$\begin{aligned}
\mathbb{E}\left[2^{f_N(\mathbf{Z}_1,\ldots,\mathbf{Z}_N)}|\mathbf{Z}_1,\ldots,\mathbf{Z}_{N-1}\right] &\leq \mathbb{E}[1+f_N(\mathbf{Z}_1,\ldots,\mathbf{Z}_N)|\mathbf{Z}_1,\ldots,\mathbf{Z}_{N-1}] \\
&\leq (1+a_N) \\
&\leq \exp(a_N).
\end{aligned}$$

Therefore

$$\mathbb{P}\left(\frac{1}{N}\sum_{i=1}^{N} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i) > t\right) \leq \exp(-Nt(\log 2) + a_N)\mathbb{E}\left[2^{\sum_{i=1}^{N-1} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i)}\right].$$

Continuing in the same way yields

$$\mathbb{P}\left(\frac{1}{N}\sum_{i=1}^{N} f_i(\mathbf{Z}_1,\ldots,\mathbf{Z}_i) > t\right) \leq \exp\left(-Nt(\log 2) + \sum_{i=1}^{N} a_i\right).$$

■

The second technical lemma (Csiszár and Narayan, 1991, Lemma 2) is a basic result about the distribution of inner product between a randomly chosen unit vector and any other fixed vector. It is a consequence of a result of Shannon (Shannon, 1959) on the distribution of the angle between a uniformly distributed unit vector and a fixed unit vector.

Lemma 17 (Lemma 2 of Csiszár and Narayan (1991)) *Let* $\mathbf{U}$ *be a random vector distributed uniformly on the unit sphere* $\mathbb{S}^{d-1}$ *in* $\mathbb{R}^d$*. Then for every unit vector* $\mathbf{u}$ *on this sphere and any* $\phi \in [(2\pi d)^{-1/2}, 1)$*, the following inequality holds:*

$$\mathbb{P}(\langle \mathbf{U}, \mathbf{u}\rangle \geq \phi) \leq (1-\phi^2)^{(d-1)/2}.$$

Lemma 18 (Packing set on the unit sphere) *Let the dimension* d *and parameter* $\phi \in [(2\pi d)^{-1/2}, 1)$ *be given. For* N *and* t *satisfying*

$$-Nt(\log 2) + N(N-1)(1-\phi^2)^{(d-1)/2} < 0 \tag{28}$$

there exists a set of $K = \lfloor (1-t)N \rfloor$ *unit vectors* $\mathcal{C}$ *such that for all distinct pairs* $\mu, \nu \in \mathcal{C}$,

$$|\langle \mu, \nu \rangle| < \phi. \tag{29}$$

Proof The goal is to generate a set of N unit vectors on the surface of the sphere $\mathbb{S}^{d-1}$ such that they have large pairwise distances or, equivalently, small pairwise inner products. To that end, define i.i.d. random variables $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_N$ uniformly distributed on $\mathbb{S}^{d-1}$ and functions

$$f_i(\mathbf{Z}_1, \ldots, \mathbf{Z}_i) = \mathbf{1}\left(\left| \langle \mathbf{Z}_i, \mathbf{Z}_j \rangle \right| > \phi, \ j < i \right).$$

That is, $f_i = 1$ if $\mathbf{Z}_i$ has large inner product with any $\mathbf{Z}_j$ for $j < i$. The conditional expectation, by a union bound and Lemma 17, is

$$\mathbb{E}\left[f_i(\mathbf{Z}_1, \ldots, \mathbf{Z}_i) | \mathbf{Z}_1, \ldots, \mathbf{Z}_{i-1} \right] \leq 2(i-1)(1-\phi^2)^{(d-1)/2}.$$

Let $a_i = 2(i-1)(1-\phi^2)^{(d-1)/2}$. Then

$$\sum_{i=1}^{N} a_i = N(N-1)(1-\phi^2)^{(d-1)/2}.$$

Then Lemma 16 shows

$$\mathbb{P}\left(\frac{1}{N} \sum_{i=1}^{N} f_i(\mathbf{Z}_1, \ldots, \mathbf{Z}_i) > t \right) \leq \exp\left(-Nt(\log 2) + N(N-1)(1-\phi^2)^{(d-1)/2} \right).$$

This inequality implies that as long as

$$-Nt(\log 2) + N(N-1)(1-\phi^2)^{(d-1)/2} < 0,$$

then there is a finite probability that $\{\mathbf{Z}_i\}$ contains a subset $\{\mathbf{Z}'_i\}$ of size $\lfloor (1-t)N \rfloor$ such that $\left| \langle \mathbf{Z}'_i, \mathbf{Z}'_j \rangle \right| < \phi$ for all (i, j). Therefore such a set exists. ■

A simple setting of the parameters gives the packing in Lemma 11.

Proof Applying Lemma 18 yields a set of K vectors $\mathcal{C}$ satisfying (28) and (29). To get a simple bound that's easy to work with, we can set

$$-Nt(\log 2) + N(N-1)(1-\phi^2)^{(d-1)/2} = 0,$$

and find an N close to this. Setting $\psi = (1-\phi^2)^{(d-1)/2}$, and solving for N we see

$$N = 1 + \frac{t \log 2}{\psi} > \frac{t}{2\psi}.$$

Now setting $K = \frac{t(1-t)}{2\psi}$ and $t = 1/2$ gives (12). So there exists a set of K vectors on $\mathbb{S}^{d-1}$ whose pairwise inner products are smaller than ϕ. ■

The maximum set of points that can be selected on a sphere of dimension d such that their pairwise inner products are bounded by ϕ is an open question. These sets are sometimes referred to as spherical codes (Conway and Sloane, 1998) because they correspond to a set of signaling points of dimension d that can be perfectly decoded over a channel with bounded noise. The bounds here are from a probabilistic construction and can be tightened for smaller d. However, in terms of scaling with d this construction is essentially optimal (Shannon, 1959).

References

Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. *SIGMOD Record*, 29(2):439–450, 2000. ISSN 0163-5808. doi: 10.1145/335191.335438. URL http://dx.doi.org/10.1145/335191.335438.

Kevin Bache and Moshe Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Keith M. Ball. An elementary introduction to modern convex geometry. In S. Levy, editor, *Flavors of Geometry*, volume 31 of *Mathematical Sciences Research Institute Publications*, pages 1–58. Cambridge University Press, 1997.

Boaz Barak, Kamalika Chaudhuri, Cynthia Dwork, Satyen Kale, Frank McSherry, and Kunal Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *Proceedings of the Twenty-Sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '07)*, pages 273–282, New York, NY, USA, 2007. ACM. doi: 10.1145/1265530.1265569. URL http://dx.doi.org/10.1145/1265530.1265569.

Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. The Johnson-Lindenstrauss Transform itself preserves differential privacy. In *IEEE 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 410–419, October 2012. doi: 10.1109/FOCS.2012.67. URL http://dx.doi.org/10.1109/FOCS.2012.67.

Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the SuLQ framework. In *Proceedings of the Twenty-Fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '05)*, pages 128–138, New York, NY, USA, 2005. ACM. doi: 10.1145/1065167.1065184. URL http://dx.doi.org/10.1145/1065167.1065184.

Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to non-interactive database privacy. In R. E. Ladner and C. Dwork, editors, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC '08)*, pages 609–618, New York, NY, USA, 2008. ACM. doi: 10.1145/1374376.1374464. URL http://dx.doi.org/10.1145/1374376.1374464.

Stephen P. Brooks. Markov chain Monte Carlo method and its application. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 47(1):69–100, April 1998. ISSN 00390526. doi: 10.1111/1467-9884.00117. URL http://dx.doi.org/10.1111/1467-9884.00117.

Stephen P. Brooks and Andrew Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7(4):434–455, December 1998. doi: 10.2307/1390675. URL http://dx.doi.org/10.2307/1390675.

Stephen P. Brooks and Gareth O. Roberts. Convergence assessment techniques for Markov chain Monte Carlo. *Statistics and Computing*, 8(4):319–335, December 1998. doi: 10.1023/A:1008820505350. URL http://dx.doi.org/10.1023/A:1008820505350.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Kamalika Chaudhuri and Daniel Hsu. Sample complexity bounds for differentially private learning. In Sham Kakade and Ulrike von Luxburg, editors, *Proceedings of the 24th Annual Conference on Learning Theory (COLT '11)*, volume 19 of *JMLR Workshop and Conference Proceedings*, pages 155–186, Budapest, Hungary, June 2011. URL http://www.jmlr.org/proceedings/papers/v19/chaudhuri11a/chaudhuri11a.pdf.

Kamalika Chaudhuri and Daniel Hsu. Convergence rates for differentially private statistical estimation. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 1327–1334, New York, NY, USA, July 2012. Omnipress. URL http://icml.cc/2012/papers/663.pdf.

Kamalika Chaudhuri and Nina Mishra. When random sampling preserves privacy. In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 198–213, Berlin, Heidelberg, August 2006. Springer-Verlag. doi: 10.1007/11818175_12. URL http://dx.doi.org/10.1007/11818175_12.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12:1069–1109, March 2011. URL http://jmlr.csail.mit.edu/papers/v12/chaudhuri11a.html.

Kamalika Chaudhuri, Anand D. Sarwate, and Kaushik Sinha. Near-optimal differentially private principal components. In P. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 998–1006, 2012. URL http://books.nips.cc/papers/files/nips25/NIPS2012_0482.pdf.

Yasuko Chikuse. *Statistics on Special Manifolds*. Number 174 in Lecture Notes in Statistics. Springer, New York, 2003.

John H. Conway and Neil J. A. Sloane. *Sphere Packing, Lattices and Groups*. Springer-Verlag, New York, 1998.

Mary Kathryn Cowles and Bradley P. Carlin. Markov Chain Monte Carlo convergence diagnostics: A comparative review. *Journal of the American Statistical Association*, 91(434):883, June 1996. ISSN 01621459. doi: 10.2307/2291683. URL http://dx.doi.org/10.2307/2291683.

Imre Csiszár and Prakash Narayan. The capacity of the arbitrarily varying channel revisited : Positivity, constraints. *IEEE Transactions on Information Theory*, 34(2):181–193, 1988. doi: 10.1109/18.2627. URL http://dx.doi.org/10.1109/18.2627.

Imre Csiszár and Prakash Narayan. Capacity of the Gaussian arbitrarily varying channel. *IEEE Transactions on Information Theory*, 37(1):18–26, 1991. doi: 10.1109/18.61125. URL http://dx.doi.org/10.1109/18.61125.

Randal Douc, Eric Moulines, and Jeffrey S. Rosenthal. Quantitative bounds on convergence of time-inhomogeneous Markov chains. *The Annals of Applied Probability*, 14(4):1643–1665, November

2004. ISSN 1050-5164. doi: 10.1214/105051604000000620. URL http://dx.doi.org/10.1214/105051604000000620.

Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC '09)*, pages 371–380, New York, NY, USA, 2009. ACM. doi: 10.1145/1536414.1536466. URL http://dx.doi.org/10.1145/1536414.1536466.

Cynthia Dwork and Adam Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):135–154, 2009. URL http://repository.cmu.edu/jpc/vol1/iss2/2.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503, Berlin, Heidelberg, 2006a. Springer-Verlag. doi: 10.1007/11761679_29. URL http://dx.doi.org/10.1007/11761679_29.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284, Berlin, Heidelberg, March 4–7 2006b. Springer. doi: 10.1007/11681878_14. URL http://dx.doi.org/10.1007/11681878_14.

Salaheddine El Adlouni, Anne-Catherine Favre, and Bernard Bobée. Comparison of methodologies to assess the convergence of Markov chain Monte Carlo methods. *Computational Statistics & Data Analysis*, 50(10):2685–2701, June 2006. ISSN 01679473. doi: 10.1016/j.csda.2005.04.018. URL http://dx.doi.org/10.1016/j.csda.2005.04.018.

Alexandre Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 211–222, 2003. doi: 10.1145/773153.773174. URL http://dx.doi.org/10.1145/773153.773174.

Arik Friedman and Assaf Schuster. Data mining with differential privacy. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*, pages 493–502, New York, NY, USA, 2010. ACM. doi: 10.1145/1835804.1835868. URL http://dx.doi.org/10.1145/1835804.1835868.

Benjamin C. M. Fung, Ke Wang, Rui Chen, and Philip S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):14:1–14:53, June 2010. doi: 10.1145/1749603.1749605. URL http://dx.doi.org/10.1145/1749603.1749605.

Srivatsava Ranjit Ganta, Shiva Prasad Kasiviswanathan, and Adam Smith. Composition attacks and auxiliary information in data privacy. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*, pages 265–273, New York, NY, USA, 2008. ACM. doi: 10.1145/1401890.1401926. URL http://dx.doi.org/10.1145/1401890.1401926.

Shuguo Han, Wee Keong Ng, and P.S. Yu. Privacy-preserving singular value decomposition. In *Proceedings of the 25th IEEE International Conference on Data Engineering (ICDE)*, pages 1267 –1270, 2009. doi: 10.1109/ICDE.2009.217. URL http://dx.doi.org/10.1109/ICDE.2009.217.

Moritz Hardt and Aaron Roth. Beating randomized response on incoherent matrices. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 1255–1268, New York, NY, USA, 2012. ACM. doi: 10.1145/2213977.2214088. URL http://dx.doi.org/10.1145/2213977.2214088.

Moritz Hardt and Aaron Roth. Beyond worst-case analysis in private singular vector computation. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13)*, pages 331–340, New York, NY, USA, June 2013. ACM. doi: 10.1145/2488608.2488650. URL http://dx.doi.org/10.1145/2488608.2488650.

Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *2009 Ninth IEEE International Conference on Data Mining (ICDM '09)*, pages 169–178, 2009. doi: 10.1109/ICDM.2009.11. URL http://dx.doi.org/10.1109/ICDM.2009.11.

Peter D. Hoff. Simulation of the matrix Bingham-von Mises-Fisher distribution, with applications to multivariate and relational data. *Journal of Computational and Graphical Statistics*, 18(2): 438–456, 2009. ISSN 1061-8600. doi: 10.1198/jcgs.2009.07177. URL http://dx.doi.org/10.1198/jcgs.2009.07177.

Galin L. Jones and James P. Hobart. Honest exploration of intractable probability distributions via Markov Chain Monte Carlo. *Statistical Science*, 16(4):312–334, 2001. doi: 10.1214/ss/1015346317. URL http://dx.doi.org/10.1214/ss/1015346317.

Galin L. Jones and James P. Hobart. Sufficient burn-in for Gibbs samplers for a hierarchical random effects model. *The Annals of Statistics*, 32(2):784–817, April 2004. doi: 10.1214/009053604000000184. URL http://dx.doi.org/10.1214/009053604000000184.

Boštjan Kaluža, Violeta Mirchevska, Erik Dovgan, Mitja Luštrek, and Matjaž Gams. An agent-based approach to care in independent living. In B. de Ruyter et al., editor, *International Joint Conference on Ambient Intelligence (AmI-10)*, volume 6439/2010 of *Lecture Notes in Computer Science*, pages 177–186. Springer-Verlag, Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-16917-5_18. URL http://dx.doi.org/10.1007/978-3-642-16917-5_18.

Mikhail Kapralov and Kunal Talwar. On differentially private low rank approximation. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '13)*, pages 1395–1414, New Orleans, LA, USA, January 2013.

Shiva Prasad Kasiviswanathan and Adam Smith. A note on differential privacy: Defining resistance to arbitrary side information. Technical Report arXiv:0803.3946v1 [cs.CR], ArXiV, March 2008. URL http://arxiv.org/abs/0803.3946.

Krishnaram Kenthapadi, Aleksandra Korolova, Ilya Mironov, and Nina Mishra. Privacy via the Johnson-Lindenstrauss transform. *Journal of Privacy and Confidentiality*, 5(1):39–71, 2013. URL http://repository.cmu.edu/jpc/vol5/iss1/2.

John E. Kolassa. Convergence and accuracy of Gibbs sampling for conditional distributions in generalized linear models. *The Annals of Statistics*, 27(1):129–142, 1999. doi: 10.1214/aos/1018031104. URL http://dx.doi.org/10.1214/aos/1018031104.

John E. Kolassa. Explicit bounds for geometric covergence of Markov chains. *Journal of Applied Probability*, 37(3):642–651, 2000. doi: 10.1239/jap/1014842825. URL http://dx.doi.org/10.1239/jap/1014842825.

Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. Closeness: A new privacy measure for data publishing. *IEEE Transactions on Knowledge and Data Engineering*, 22(7):943–956, 2010. doi: 10.1109/TKDE.2009.139. URL http://dx.doi.org/10.1109/TKDE.2009.139.

Kun Liu, Hillol Kargupta, and Jessica Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):92–106, 2006. doi: 10.1109/TKDE.2006.14. URL http://dx.doi.org/10.1109/TKDE.2006.14.

Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE)*, page 24, 2006. doi: 10.1109/ICDE.2006.1. URL http://dx.doi.org/10.1109/ICDE.2006.1.

Ashwin Machanavajjhala, Daniel Kifer, John M. Abowd, Johannes Gehrke, and Lars Vilhuber. Privacy: Theory meets practice on the map. In *IEEE 24th International Conference on Data Engineering (ICDE)*, pages 277–286, April 2008. doi: 10.1109/ICDE.2008.4497436. URL http://dx.doi.org/10.1109/ICDE.2008.4497436.

Frank McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD Conference*, pages 19–30, 2009. doi: 10.1145/1559845.1559850. URL http://dx.doi.org/10.1145/1559845.1559850.

Frank McSherry and Ilya Mironov. Differentially private recommender systems: Building privacy into the netflix prize contenders. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data (KDD)*, pages 627–636, 2009. doi: 10.1145/1557019.1557090. URL http://dx.doi.org/10.1145/1557019.1557090.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS '07)*, pages 94–103, October 2007. doi: 10.1109/FOCS.2007.41. URL http://dx.doi.org/10.1109/FOCS.2007.41.

Ilya Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS '12)*, pages 650–661, New York, NY, USA, 2012. ACM. doi: 10.1145/2382196.2382264. URL http://dx.doi.org/10.1145/2382196.2382264.

Noman Mohammed, Rui Chen, Benjamin C. M. Fung, and Philip S. Yu. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*, pages 493–501, New York, NY, USA,

2011. ACM. doi: 10.1145/2020408.2020487. URL http://dx.doi.org/10.1145/2020408.2020487.

Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing (STOC '07)*, pages 75–84, New York, NY, USA, 2007. ACM. doi: 10.1145/1250790.1250803. URL http://dx.doi.org/10.1145/1250790.1250803.

Gareth O. Roberts. Bounds on regeneration times and convergence rates for Markov chains. *Stochastic Processes and their Applications*, 80(2):211–229, April 1999. ISSN 03044149. doi: 10.1016/S0304-4149(98)00085-4. URL http://dx.doi.org/10.1016/S0304-4149(98)00085-4.

Gareth O. Roberts and Sujit K. Sahu. Approximate predetermined convergence properties of the Gibbs sampler. *Journal of Computational and Graphical Statistics*, 10(2):216–229, June 2001. ISSN 1061-8600. doi: 10.1198/10618600152627915. URL http://dx.doi.org/10.1198/10618600152627915.

Jeffrey S. Rosenthal. Minorization conditions and convergence rates for Markov Chain Monte Carlo. *Journal of the American Statistical Association*, 90(430):558–566, June 1995. ISSN 01621459. doi: 10.2307/2291067. URL http://dx.doi.org/10.2307/2291067.

Benjamin I. P. Rubinstein, Peter L. Bartlett, Ling Huang, and Nina Taft. Learning in a large function space: Privacy-preserving mechanisms for SVM learning. *Journal of Privacy and Confidentiality*, 4(1):65–100, 2012. URL http://repository.cmu.edu/jpc/vol4/iss1/4/.

Claude. E. Shannon. Probability of error for optimal codes in a Gaussian channel. *Bell System Technical Journal*, 38:611–656, 1959.

Adam Smith. Privacy-preserving statistical estimation with optimal convergence rates. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC '11)*, pages 813–822, New York, NY, USA, 2011. ACM. doi: 10.1145/1993636.1993743. URL http://dx.doi.org/10.1145/1993636.1993743.

Gilbert W. Stewart. On the early history of the singular value decomposition. *SIAM Review*, 35(4):551–566, December 1993.

Latanya Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, October 2002. doi: 10.1142/S0218488502001648. URL http://dx.doi.org/10.1142/S0218488502001648.

Peter van der Putten and Maarten van Someren. CoIL Challenge 2000: The Insurance Company Case, 2000. URL http://www.liacs.nl/~putten/library/cc2000/. Leiden Institute of Advanced Computer Science Technical Report 2000-09.

Larry Wasserman and Shuheng Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375–389, 2010. doi: 10.1198/jasa.2009.tm08651. URL http://dx.doi.org/10.1198/jasa.2009.tm08651.

Oliver Williams and Frank McSherry. Probabilistic inference and differential privacy. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2451–245, 2010. URL `http://books.nips.cc/papers/files/nips23/NIPS2010_1276.pdf`.

Bin Yu. Assouad, Fano, and Le Cam. In David Pollard, Erik Torgersen, and Grace L. Yang, editors, *Festschrift for Lucien Le Cam*, Research Papers in Probability and Statistics, chapter 29, pages 423–425. Springer-Verlag, 1997.

Justin Z. Zhan and Stan Matwin. Privacy-preserving support vector machine classification. *International Journal of Intelligent Information and Database Systems*, 1(3/4):356–385, 2007. doi: 10.1504/IJIIDS.2007.016686.

Shuheng Zhou, Katrina Ligett, and Larry Wasserman. Differential privacy with compression. In *Proceedings of the 2009 International Symposium on Information Theory (ISIT)*, pages 2718–2722, Seoul, South Korea, June–July 2009. doi: 10.1109/ISIT.2009.5205863.

Journal of Machine Learning Research 14 (2013) 2945-2977 Submitted 8/11; Revised 4/12; Published 10/13

Parallel Vector Field Embedding

Binbin Lin BINBINLIN@ZJU.EDU.CN
Xiaofei He XIAOFEIHE@CAD.ZJU.EDU.CN
Chiyuan Zhang PLUSKID@GMAIL.COM
State Key Lab of CAD&CG
College of Computer Science
Zhejiang University
Hangzhou, 310058, China

Ming Ji MINGJI1@ILLINOIS.EDU
Department of Computer Science
University of Illinois at Urbana Champaign
Urbana, IL 61801, USA

Editor: Mikhail Belkin

Abstract

We propose a novel local isometry based dimensionality reduction method from the perspective of vector fields, which is called parallel vector field embedding (PFE). We first give a discussion on local isometry and global isometry to show the intrinsic connection between parallel vector fields and isometry. The problem of finding an isometry turns out to be equivalent to finding orthonormal parallel vector fields on the data manifold. Therefore, we first find orthonormal parallel vector fields by solving a variational problem on the manifold. Then each embedding function can be obtained by requiring its gradient field to be as close to the corresponding parallel vector field as possible. Theoretical results show that our method can precisely recover the manifold if it is isometric to a connected open subset of Euclidean space. Both synthetic and real data examples demonstrate the effectiveness of our method even if there is heavy noise and high curvature.

Keywords: manifold learning, isometry, vector field, covariant derivative, out-of-sample extension

1. Introduction

In many data analysis tasks, one is often confronted with very high dimensional data. There is a strong intuition that the data may have a lower dimensional intrinsic representation. Various researchers have considered the case when the data is sampled from a submanifold embedded in much higher dimensional Euclidean space. Consequently, estimating and extracting the low dimensional manifold structure, or specifically the intrinsic topological and geometrical properties of the data manifold, become a crucial problem. These problems are often referred to as *manifold learning* (Belkin and Niyogi, 2007).

The most natural technique to exact low dimensional manifold structure with given finite samples is dimensionality reduction. The early work for dimensionality reduction includes principal component analysis (PCA, Jolliffe, 1989), multidimensional scaling (MDS, Cox and Cox, 1994) and linear discriminant analysis (LDA, Duda et al., 2000). PCA is probably the most popular dimensionality reduction methods. Given a data set, PCA finds the directions along which the data

has maximum variance. However, these linear methods may fail to recover the intrinsic manifold structure when the data manifold is not a low dimensional subspace or an affine manifold.

There are various works on nonlinear dimensionality reduction in the last decade. The typical work includes isomap (Tenenbaum et al., 2000), locally linear embedding (LLE, Roweis and Saul, 2000), Laplacian eigenmaps (LE, Belkin and Niyogi, 2001), Hessian eigenmaps (HLLE, Donoho and Grimes, 2003) and diffusion maps (Coifman and Lafon, 2006; Lafon and Lee, 2006; Nadler et al., 2006). Isomap generalizes MDS to the nonlinear manifold case which tries to preserve pairwise geodesic distances on the data manifold. Diffusion maps tries to preserve another meaningful distance, that is, diffusion distance on the manifold. Isomap is an instance of global isometry based dimensionality reduction techniques, which tries to preserve the distance function or the metric of the manifold globally. One limitation of Isomap is that it requires the manifold to be geodesically convex. HLLE is based on local isometry criterion, which successfully overcomes this problem. Laplacian operator and Hessian operator are two of the most important differential operators in manifold learning. Intuitively, Laplacian measures the smoothness of the functions, while Hessian measures how a function changes the metric of the manifold. However, the Laplacian based methods like LLE and LE mainly focus on the smoothness of the embedding function, which may not be an isometry. The major difficulty in Hessian based methods is that they have to estimate the second order derivative of embedding functions, and consequently they have strong requirement on data samples.

One natural nonlinear extension of PCA is kernel principal component analysis (kernel PCA, Schölkopf et al., 1998). Interestingly, Ham et al. (2004) showed that Isomap, LLE and LE are all special cases of kernel PCA with specific kernels. Recently, Maximum Variance Unfolding (MVU, Weinberger et al., 2004) is proposed to learn a kernel matrix that preserves pairwise distances on the manifold. MVU can be thought of as an instance of local isometry with additional consideration that the distances between two points that are not neighbors are maximized.

Tangent space based methods have also received considerable interest recently, such as local tangent space alignment (LTSA, Zhang and Zha, 2004), manifold charting (Brand, 2003), Riemannian Manifold Learning (RML, Lin and Zha, 2008) and locally smooth manifold learning (LSML, Dollár et al., 2007). These methods try to find coordinates representation for curved manifolds. LTSA tries to construct a global coordinate via local tangent space alignment. Manifold charting has a similar strategy, which tries to expand the manifold by splicing local charts. RML uses normal coordinate to unfold the manifold, which aims to preserve the metric of the manifold. LSML tries to learn smooth tangent spaces of the manifold by proposing a smoothness regularization term of tangent spaces. Vector diffusion maps (VDM, Singer and Wu, 2011) is a much recent work which considers the tangent spaces structure of the manifold to define and preserve the vector diffusion distance.

In this paper, we propose a novel dimensionality reduction method, called parallel vector field embedding (PFE), from the perspective of vector fields. The theory of vector fields is a basic tool for discovering the geometry and topology of the manifold. We first give a discussion on local isometry and global isometry to show the intrinsic connection between parallel vector fields and isometry. The problem of finding an isometry turns out to be equivalent to finding orthonormal parallel vector fields on the data manifold. Therefore, we first find orthonormal parallel vector fields by minimizing the covariant derivative of a vector field. We then find an embedding function whose gradient field is as close to the parallel field as possible. In this way, the obtained embedding function would vary linearly along the geodesics of the manifold. Naturally, the corresponding embedding consisted

of embedding functions preserves the metric of the manifold. As pointed out by Goldberg et al. (2008), almost all spectral methods including LLE, LE, LTSA and HLLE use global normalization for embedding, which sacrifices isometry. PFE overcomes this problem by normalizing vector fields locally. Our theoretical study shows that, if the manifold is isometric to a connected open subset of Euclidean space, our method can faithfully recover the metric structure of the manifold.

The organization of the paper is as follows: In the next section, we provide a description of the dimensionality reduction problem from the perspectives of isometry and vector fields. In Section 3, we introduce our proposed Parallel Field Embedding algorithm. The extensive experimental results on both synthetic and real data sets are presented in Section 4. Finally, we provide some concluding remarks and suggestions for future work in Section 5.

2. Dimensionality Reduction from Geometric Perspective

Let $(\mathcal{M}, g)$ be a d-dimensional Riemannian manifold embedded in a much higher dimensional Euclidean space $\mathbb{R}^m$, where g is a Riemannian metric on $\mathcal{M}$. A Riemannian metric is a Euclidean inner product g_p on each of the tangent space $T_p\mathcal{M}$, where p is a point on the manifold $\mathcal{M}$. In addition we assume that g_p varies smoothly (Petersen, 1998). This means that for any two smooth vector fields X, Y the inner product $g_p(X_p, Y_p)$ should be a smooth function of p. The subscript p will be suppressed when it is not needed. Thus we might write $g(X,Y)$ or $g_p(X,Y)$ with the understanding that this is to be evaluated at each p where X and Y are defined. Generally we use the induced metric for $\mathcal{M}$. That is, the inner product defined in the tangent space of $\mathcal{M}$ is the same as that in the ambient space $\mathbb{R}^m$, that is, $g(u,v) = \langle u, v\rangle$ where $\langle\cdot,\cdot\rangle$ denote the canonical inner product in $\mathbb{R}^m$. In the problem of dimensionality reduction, one tries to find a smooth map: $F : \mathcal{M} \to \mathbb{R}^d$, which preserves the topological and geometrical properties of $\mathcal{M}$.

However, for some kinds of manifolds, it is impossible to preserve all the geometrical and topological properties. For example, consider a two-dimensional sphere, there is no such map that maps the sphere to a plane without breaking the topology of the sphere. Thus, there should be some assumptions of the data manifold. In this paper, we consider a relatively general assumption that the manifold $\mathcal{M}$ is diffeomorphic to an open subset of the Euclidean space $\mathbb{R}^d$. In other words, we assume that there exists a topology preserving map from $\mathcal{M}$ to $\mathbb{R}^d$.

Definition 1 (Diffeomorphism, Lee, 2003) *A diffeomorphism between manifolds $\mathcal{M}$ and $\mathcal{N}$ is a smooth map $F : \mathcal{M} \to \mathcal{N}$ that has a smooth inverse. We say $\mathcal{M}$ and $\mathcal{N}$ are diffeomorphic if there exists a diffeomorphism between them.*

For example, there is a diffeomorphism between a semi-sphere and a subset of $\mathbb{R}^2$. However, there is no diffeomorphism between a sphere and any subset of $\mathbb{R}^2$. In this paper, we only consider manifolds that are diffeomorphic to an open connected subset of Euclidean space like semi-sphere, swiss roll, swiss roll with hole, and so on.

2.1 Local Isometry and Global Isometry

With the assumption that the manifold is diffeomorphic to an open subset of $\mathbb{R}^d$, the goal of dimensionality reduction is to preserve the intrinsic geometry of the manifold as much as possible. Ideally, one hopes to preserve the metric of the manifold, or intuitively the pairwise distance between data points. This leads to the concept of isometry. Here we consider two kinds of isometry, that are,

local isometry and global isometry.[1] In the following we give the definitions and properties of local isometry and global isometry.

Definition 2 (Local Isometry, Lee, 2003) *Let $(\mathcal{M},g)$ and $(\mathcal{N},h)$ be two Riemannian manifolds, where g and h are metrics on them. For a map between manifolds $F:\mathcal{M}\to\mathcal{N}$, F is called local isometry if $h(dF_p(v),dF_p(v))=g(v,v)$ for all $p\in\mathcal{M}, v\in T_p\mathcal{M}$. Here dF is the differential of F.*

dF is also known as *pushforward* and denoted as F_* in many texts. For a fixed point $p\in\mathcal{M}$, dF is a linear map between $T_p\mathcal{M}$ and the corresponding tangent space $T_{F(p)}\mathcal{N}$. According to the definition, dF preserves the norm of tangent vectors. Moreover, we have the following theorem:

Theorem 1 (Petersen, 1998) *Let $F:\mathcal{M}\to\mathcal{N}$ be a local isometry, then*

1. *F maps geodesics to geodesics.*
2. *F is distance decreasing.*
3. *if F is also a bijection, then it is distance preserving.*

Intuitively, local isometry preserves the metric of the manifold locally. If the local isometry F is also a diffeomorphism, then it becomes global isometry.

Definition 3 (Global Isometry, Lee, 2003) *A map $F:\mathcal{M}\to\mathcal{N}$ is called global isometry between manifolds if it is a diffeomorphism and also a local isometry.*

If F is a global isometry, then its inverse F^{-1} is also a global isometry. We have the following proposition.

Proposition 1 *A global isometry preserves geodesics. If $F:\mathcal{M}\to\mathcal{N}$ is a global isometry, then for any two points $p,q\in\mathcal{M}$, we have $d(p,q)=d(F(p),F(q))$, where $d(\cdot,\cdot)$ denotes geodesic distance between two points.*

Proof For any two points $p,q\in\mathcal{M}$, we have $d(p,q)\geq d(F(p),F(q))$ according to the third statement of Theorem 1. Since $F^{-1}:\mathcal{N}\to\mathcal{M}$ is also a global isometry, we have $d(p,q)\leq d(F(p),F(q))$. Thus $d(p,q)=d(F(p),F(q))$. ■

Clearly, we hope the map F is a global isometry. This is because that, although local isometry maps geodesics to geodesics, the shortest geodesic between two points on $\mathcal{M}$ may not be the shortest geodesic on $\mathcal{N}$. Please see Figure 1 as an illustrative example. Clearly the map F is a local isometry. However, consider two points p and q on $\mathcal{M}$, we have $d(p,q)>d(F(p),F(q))$. Therefore F is not a global isometry.

It is usually very difficult to find a global isometry. Isomap is designed to find the global isometry. However, it is known that the computational cost is very expensive since pairwise distances have to be estimated. Also, it has been shown that Isomap cannot handle geodesically non-convex manifolds where in that case the geodesic distances cannot be accurately estimated. On the other hand, based on our assumption that the manifold $\mathcal{M}$ is diffeomorphic to an open subset of $\mathbb{R}^d$, it suffices to find a local isometry which is also a diffeomorphism, according to Definition 3.

1. In many differential geometry textbooks, global isometry is often referred to as *isometry* or *Riemannian isometry.*

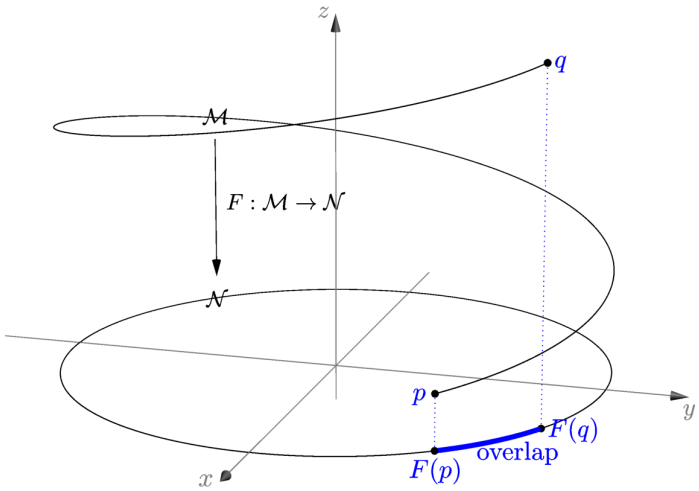


Figure 1: Local isometry but not global isometry. F is a map from $\mathcal{M}$ to $\mathcal{N}$. Clearly F is a local isometry. However, due to the overlap, it is not a global isometry.

2.2 Gradient Fields and Local Isometry

Our analysis has shown that finding a global isometry is equivalent to finding a local isometry which is also a diffeomorphism. Given a map $F = (f_1, \ldots, f_d) : \mathcal{M} \to \mathbb{R}^d$, there is a deep connection between local isometry and the differential $dF = (df_1, \ldots, df_d)$. For a function f on the manifold $f : \mathcal{M} \to \mathbb{R}$, we will not strictly distinguish between its *differential* df and its *gradient field* ∇f in this paper. Actually, they are dual 1-form which is uniquely determined by each other once the metric of the manifold is given. For the relationship between local isometry and differential, we have the following proposition:

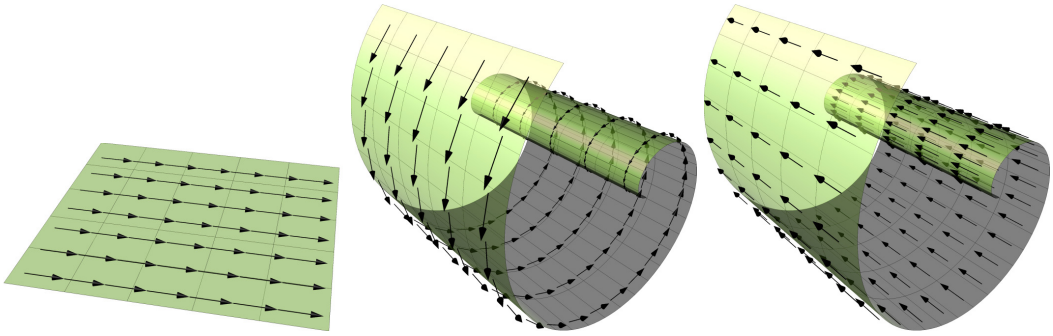
Proposition 2 *Consider a map $F : \mathcal{M} \subset \mathbb{R}^m \to \mathbb{R}^d$. Let $f_i, i = 1, \ldots, d$ denote the component of F which maps the manifold to $\mathbb{R}$, that is, $F = (f_1, \ldots, f_d)$. The following three statements are equivalent:*

1. *F is a local isometry.*
2. *dF_p is an orthogonal transformation for all $p \in \mathcal{M}$.*
3. *$\langle df_i, df_j \rangle_p = \delta_{ij}, i, j = 1, ..., d, \forall p \in \mathcal{M}$.*

Proof $2 \Leftrightarrow 3$ is trivial by the definition of orthogonal transformation. $2 \Rightarrow 1$ is obvious. Next we prove $1 \Rightarrow 2$. Since we use the induced metric for the manifold $\mathcal{M}$, the computation of inner product in tangent space is the same as the standard inner product in Euclidean space. We have $g(u, v) = \langle u, v \rangle, \forall u, v \in T_p\mathcal{M}$. According to Definition 2, we have $\langle u, u \rangle = \langle dF_p(u), dF_p(u) \rangle, \forall p \in \mathcal{M}, u \in T_p\mathcal{M}$. For arbitrary vectors u and $v \in T_p\mathcal{M}$, then we have

$$\langle dF_p(u+v), dF_p(u+v) \rangle = \langle u+v, u+v \rangle.$$

By expanding both side and notice that $\langle dF_p(u), dF_p(u) \rangle = \langle u, u \rangle$ and $\langle dF_p(v), dF_p(v) \rangle = \langle v, v \rangle$, we have $\langle dF_p(u), dF_p(v) \rangle = \langle u, v \rangle$ which implies that dF is an orthogonal transformation. ■



(a) $\mathbb{R}^2$ (b) Swiss Roll (c) Swiss Roll

Figure 2: Examples of parallel fields. The parallel fields on Euclidean space are all constant vector fields.

This proposition indicates that finding a local isometry F is equivalent to finding d orthonormal differentials $df_i, i = 1,...,d$, or gradient fields $\nabla f_i, i = 1,...,d$ since we have $\langle \nabla f_i, \nabla f_j \rangle = \langle df_i, df_j \rangle = \delta_{ij}$.

3. Parallel Field Embedding

In this section, we introduce our parallel field embedding (PFE) algorithm for dimensionality reduction.

Our goal is to find a map $F = (f_1, \ldots, f_d) : \mathcal{M} \subset \mathbb{R}^m \to \mathbb{R}^d$ which preserves the metric of the manifold. According to Proposition 2, its differential (or gradient fields) should be orthonormal. In the next subsection, we show that if such differential exists, then each df_i (or ∇f_i) has to be *parallel*, that is $\nabla df_i = 0$ (or $\nabla\nabla f_i = 0$). Naturally, we propose a vector field based method for solving this problem. We first try to find orthonormal parallel vector fields on the manifold. Then we try to reconstruct the map whose gradient fields can best approximate the parallel fields. In Theorem 2, we show that if the manifold can be isometrically embedded into the Euclidean space, then there exist orthonormal parallel fields and each parallel field is exactly a gradient field. Therefore, in such cases our proposed approach can find the optimal embedding which is a global isometry.

3.1 Parallel Vector Fields

In this subsection, we will show the properties of parallel fields. We will also discuss the relationship among local isometry, global isometry and parallel fields.

Definition 4 (Parallel Field, Petersen, 1998) *A vector field X on manifold $\mathcal{M}$ is a parallel vector field (or parallel field) if*

$$\nabla X \equiv 0,$$

where ∇ is the covariant derivative on the manifold $\mathcal{M}$.

Figure 2 shows some examples of parallel fields on Euclidean space and Swiss Roll. Given a point p on the manifold and a vector v_p on the tangent space $T_p\mathcal{M}$, then $\nabla_{v_p} X$ is a vector at point p which measures how the vector field X changes along the direction v_p at point p. Since p is arbitrary, given

a vector field Y, then $\nabla_Y X$ is a vector field which measures how the vector field X changes along the vector field Y on the manifold. Since $\nabla X : Y \mapsto \nabla_Y X$ is a map which maps a vector field Y to another vector field $\nabla_Y X$, $\nabla X \equiv 0$ also means that for any vector field Y on the manifold, we have $\nabla_Y X = 0$ and vice versa. For parallel fields, we also have the following proposition:

Proposition 3 *Let V and W be parallel fields on $\mathcal{M}$ associated with the metric g. We define a function $h : \mathcal{M} \to \mathbb{R}$ as follows:*

$$h(p) = g_p(V,W),$$

where g_p represents the inner product at p. Then $h(p) =$ constant, $\forall p \in \mathcal{M}$.

Proof Since V and W are parallel fields, we have $\nabla V = \nabla W = 0$ or $\nabla_Y V = \nabla_Y W = 0$ for any vector field Y.

We first show that a vector field is a derivation. For simplicity, let v_p be a tangent vector at point p on Euclidean space $\mathbb{R}^m$. Then v_p defines the directional derivative in the direction v_p at p as follows:

$$v_p f = D_v f(p) = \frac{d}{dt}|_{t=0} f(p+tv).$$

For tangent vectors on a general manifold, we define them as derivations which satisfies the Leibniz's rule, that is, $v_p(fg) = f(p)v_p g + g(p)v_p f$. If p varies, then all these v_ps constitute a vector field. Since for each point p, v_p is a derivation at p. Then the vector field is a derivation on the manifold. Let X be an arbitrary smooth vector field, then we apply it to the function h and we have

$$\begin{aligned} X(h) &= Xg(V,W) \\ &= g(\nabla_X V, W) + g(V, \nabla_X W) \\ &= 0 + 0 = 0. \end{aligned}$$

The second equation is due to the property of the covariant derivative. Since the above equation holds for arbitrary X, we have $h(p) = g_p(V,W) =$ constant. ■

Corollary 1 *Let V and W be parallel fields on $\mathcal{M}$ associated with the metric g, then $\int_{\mathcal{M}} g(V,W)dx = 0$ if and only if $\forall\ p \in \mathcal{M}$, $g_p(V,W) = 0$.*

Proof From Proposition 3, we see that $g(V,W) =$ constant. Thus, $\int_{\mathcal{M}} g(V,W)dx = \text{vol}(\mathcal{M})g(V,W)$. Since $\text{vol}(\mathcal{M}) > 0$, $\int_{\mathcal{M}} g(V,W)dx = 0$ if and only if $g(V,W) = 0$ or $\forall p \in \mathcal{M}$, $g_p(V,W) = 0$. ■

This corollary tells us if we want to check the orthogonality of the parallel fields at every point, it suffices to compute the integral of the inner product of the parallel fields. This is much more convenient for finding orthogonal parallel fields.

Also we have the following corollary:

Corollary 2 *Let V be a parallel vector field on $\mathcal{M}$, then $\forall p \in \mathcal{M}, \|V_p\| =$ constant where V_p represents the vector at p of the vector field V.*

Proof Let $W = V$ in Proposition 3, then $\forall p \in \mathcal{M}$, we have $\|V_p\|^2 = g_p(V,V) =$ constant. ■

Since every tangent vector of a parallel field has a constant length, we can perform normalization

of the parallel field simply as dividing every tangent vector of the parallel field by a same length. According to these results, finding orthonormal parallel fields becomes much easier: we first find orthogonal parallel fields on the manifold one by one by requiring that the integral of the inner product of two parallel fields is zero. We then normalize the vectors of parallel fields to be unit norm.

Before presenting our main result, we still need to introduce some concepts and properties on the relationship between isometry and parallel fields. We begin with the properties of the differential of a map. We have the following lemma.

Lemma 1 (Please see Lemma 3.5 in Lee, 2003) *Let $F : \mathcal{M} \to \mathcal{N}$ and let $p \in \mathcal{M}$.*

1. *$dF : T_p\mathcal{M} \to T_{F(p)}\mathcal{N}$ is linear.*
2. *If F is a diffeomorphism, then $dF : T_p\mathcal{M} \to T_{F(p)}\mathcal{N}$ is an isomorphism.*

This lemma shows that locally dF is an isomorphism if it is a diffeomorphism. Next, we show that a parallel field is uniquely determined locally.

Proposition 4 *Let $\mathcal{M}$ be an open connected manifold. For a given $p \in \mathcal{M}$, a parallel field X is uniquely determined by the vector X_p, where X_p denotes the value of the vector field X on the point p.*

Proof The equation $\nabla X \equiv 0$ is linear in X, so the space of parallel fields is a vector space. Therefore, it suffices to show that $X \equiv 0$ provided $X_p = 0$. According to Proposition. 3, a parallel field has constant length. Thus for any point $q \in \mathcal{M}$, we have $\|X_q\|^2 = \|X_p\|^2 = 0$. ■

Next we show that the differential of an isometry preserves covariant derivative.

Lemma 2 (Please see the exercise (2) in Chapter 3 of Petersen, 1998) *If $F : \mathcal{M} \to \mathcal{N}$ is a global isometry, then we have $dF(\nabla_X Y) = \nabla_{dF(X)} dF(Y)$ for all vector fields X and Y.*

More importantly, we show that an isometry preserves parallelism, that is, its differential carries a parallel vector field to another parallel vector field.

Proposition 5 *If $F : \mathcal{M} \to \mathcal{N}$ is a global isometry, then*

1. *dF maps parallel fields to parallel fields.*
2. *dF is an isometric isomorphism on the space of parallel fields.*

Proof Let Y be a parallel field on $\mathcal{M}$, we show that $dF(Y)$ is also a parallel field. It suffices to show that $\nabla_Z dF(Y) = 0$ for arbitrary Z on $\mathcal{N}$. According to Lemma 2, for any $p \in \mathcal{M}$ and any vector field X on $\mathcal{M}$, we have $\nabla_{dF(X)} dF(Y) = dF(\nabla_X Y) = 0$ hold at p. Since X is arbitrary and dF is an isomorphism (Lemma. 1) at p, then $dF(X)_p$ can be an arbitrary vector at p. Since p is also arbitrary, then all these tangent vectors $dF(X)_p$ constitute an arbitrary vector field. Thus $\nabla_Z dF(Y) = 0$ holds for arbitrary vector field Z which proves the first statement.

For the second statement, since parallel fields are determined locally (Proposition 4), we only have to show that dF is an isometrically isomorphism locally. Firstly, F is diffeomorphism. According to Lemma 1, dF is a local isomorphism. Secondly, according to the Definition 2, dF is a

local isometry. Combining these two facts, dF is an isometric isomorphism on the space of parallel fields. ■

Now we show that the gradient fields of a local isometry are also parallel fields.

Proposition 6 *If $F = (f_1, \ldots, f_d) : \mathcal{M} \to \mathbb{R}^d$ is a local isometry, then each df_i is parallel, that is, $\nabla df_i = 0$, $i = 1, \ldots, d$.*

Proof According to the property of local isometry, for very point $p \in \mathcal{M}$, there is a neighborhood $U \subset \mathcal{M}$ of p such that $F|_U : U \to F(U)$ is a global isometry of U onto an open subset $F(U)$ of $\mathbb{R}^d$ (please see Lee, 2003, pg. 187). Since $F(U)$ is an open subset of $\mathbb{R}^d$, we can choose an orthonormal basis $\partial_i, i = 1, \ldots, d$ for $F(U)$. Since $F^{-1}|_U$ is also a global isometry, thus $dF^{-1}(\partial_i)$ is an orthonormal basis of U. It can be seen as $\langle dF^{-1}(\partial_i), dF^{-1}(\partial_j)\rangle = \langle \partial_i, \partial_j \rangle = \delta_{ij}$. The first equation is due to the definition of isometry. Since $F|_U$ is a global isometry, $dF|_U$ is orthonormal with respect to these coordinates. Thus we can rewrite $F|_U$ as $F(x) = Ox + b$, where $x \in U$, O is an orthonormal matrix and $b \in \mathbb{R}^d$. Note that $dF = (df_1, \ldots, df_d)$, thus df_i has constant coefficients and we have $\nabla df_i = 0$ at each local neighborhood. Since we can choose arbitrary p, $\nabla df_i \equiv 0$ holds on the whole manifold. ■

This proposition tells us that the gradient field of a local isometry is also a parallel field. Since it is usually not easy to find a global isometry directly, in this paper, we try to find a set of orthonormal parallel fields first, and then find an embedding function whose gradient field is equal to the parallel field. Our main theorem will show that such an embedding is a global isometry.

3.2 Objective Function

As stated before, we first try to find vector fields which are as parallel as possible on the manifold. Let V be a smooth vector field on $\mathcal{M}$. By definition, the covariant derivative of V should be zero. That is, $\nabla V \equiv 0$. Naturally, we define our objective function as follows:

$$E(V) = \int_{\mathcal{M}} \|\nabla V\|_{\mathrm{HS}}^2 dx, \quad s.t., \int_{\mathcal{M}} \|V\|^2 = 1, \tag{1}$$

where $\|\cdot\|_{\mathrm{HS}}$ denotes Hilbert-Schmidt tensor norm (see Defant and Floret, 1993). The constraint removes arbitrary scale of the vector field. Once we obtain the first parallel vector field V_1, by using orthogonality constraint $\int_{\mathcal{M}} g(V_1, V_2) = 0$, we can find the second vector field V_2, and so on. After finding d orthogonal vector fields $V_1, \ldots, V_d$, we normalize the vector fields at each point:

$$\|V_i|_x\| = 1, \forall x \in \mathcal{M}, \tag{2}$$

where $V_i|_x \in T_x\mathcal{M}$ denote the vector at x of the vector field V_i.

$E(V)$ enforce the vector fields to be parallel. The norm of the tangent vector $\|V_i|_x\|$ represents the scale of the map at x, thus the normalization $\|V_i|_x\|$ removes the scale locally. This is quite different from traditional manifold learning algorithms. Traditional methods remove the scale by normalizing the norm of embedding function which is a global normalization. As pointed out by Goldberg et al. (2008), the embedding functions obtained by these traditional methods are not isometry. As we discussed in Section 2, the gradient fields of the isometry have to be orthonormal parallel fields. However, traditional manifold learning methods may not satisfy this requirement, which will be shown in our experiments.

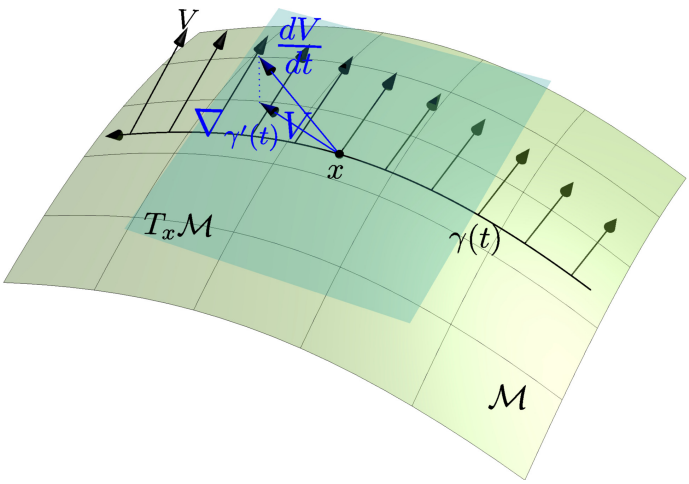


Figure 3: Covariant derivative demonstration. Let V, Y be two vector fields on manifold $\mathcal{M}$. Given a point $x \in \mathcal{M}$, let $\nabla_Y V|_x$ denote the tangent vector at x of the vector field $\nabla_Y V$, we show how to compute the vector $\nabla_Y V|_x$. Let $\gamma(t)$ be a curve on $\mathcal{M}$: $\gamma : I \to \mathcal{M}$ which satisfies $\gamma(0) = x$ and $\gamma'(0) = Y_x$. Then the covariant derivative along the direction $\frac{d\gamma(t)}{dt}|_{t=0}$ can be computed by projecting $\frac{dV}{dt}|_{t=0}$ to the tangent space $T_x\mathcal{M}$ at x. In other words, $\nabla_{\gamma'(0)} V|_x = P_x(\frac{dV}{dt}|_{t=0})$, where $P_x : v \in \mathbb{R}^m \to P_x(v) \in T_x\mathcal{M}$ is the projection matrix. It is not difficult to check that the computation of $\nabla_Y V|_x$ is independent to the choice of the curve γ.

In the following we provide some explanation of our objective function. ∇V is the covariant derivative of V that measures the change of the vector field V. If ∇V vanishes, V is a parallel vector field which we are looking for. Formally, ∇V is a $(1,1)$-tensor which maps a vector field Y to another vector field $\nabla_Y V$ and satisfies $\nabla_{\alpha Y} V = \alpha \nabla_Y V$ for any function α. For a fixed point $x \in \mathcal{M}$, let $\nabla V|_x$ denote the tensor value at x of the tensor field ∇V. It is a linear map on the tangent space $T_x\mathcal{M}$. We show what $\nabla V|_x$ is when given an orthonormal basis. Let $\partial_1, \ldots, \partial_d$ be an orthonormal basis of $T_x\mathcal{M}$, then the element of $\nabla V|_x$ would be $g_x(\nabla_{\partial_i} V, \partial_j)$ where $g_x(\cdot,\cdot)$ denote the inner product at point x. By the definition of Hilbert-Schmidt tensor norm (see Defant and Floret, 1993), we have

$$\begin{aligned} \|\nabla V|_x\|_{\mathrm{HS}}^2 &= \sum_{i=1}^{d}\sum_{j=1}^{d} (g_x(\nabla_{\partial_i} V, \partial_j))^2 \\ &= \sum_{i=1}^{d} g_x(\nabla_{\partial_i} V, \nabla_{\partial_i} V). \end{aligned} \tag{3}$$

The second equation uses the fact $g_x(\nabla_{\partial_i} V, \nabla_{\partial_i} V) = \sum_{j=1}^{d} (g_x(\nabla_{\partial_i} V, \partial_j))^2$. It is important to point out that the Hilbert-Schmidt norm $\|\nabla V\|_{\mathrm{HS}}$ is independent to the choice of the basis of tangent space. Thus our objective function $E(V)$ is well defined. According to the above equations, the computation of $\|\nabla V\|_{\mathrm{HS}}$ depends on the computation of the norm of covariant derivative $\nabla_{\partial_i} V$. Next we show the geometrical meaning of covariant derivative. For a given direction Y_x at $x \in \mathcal{M}$,

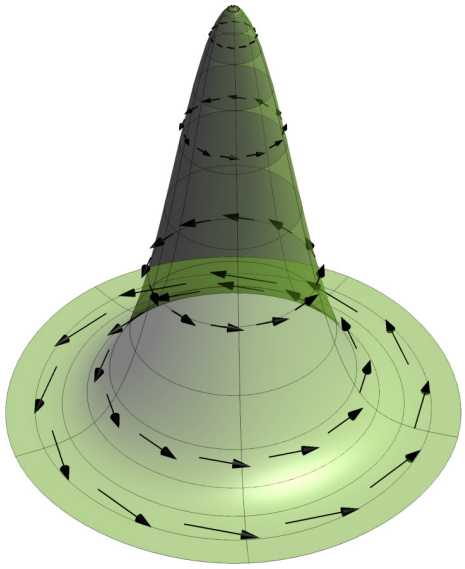


Figure 4: An example of a vector field but not a gradient field. This vector field has loops, thus it cannot be a gradient field for any function.

let $\nabla_Y V|_x$ denote the vector at x of vector field $\nabla_Y V$. Then $\nabla_Y V|_x$ is also a vector at x which is demonstrated in Figure 3.

After finding the parallel vector fields V_i on $\mathcal{M}$, the embedding function can be obtained by minimizing the following objective function:

$$\Phi(f) = \int_{\mathcal{M}} \|\nabla f - V\|^2 dx. \tag{4}$$

The solution of $\Phi(f)$ is not unique, but differs with a constant function.

In the following, we show that if the manifold is flat and diffeomorphic to an open connected subset of Euclidean space $\mathbb{R}^d$, then our method can successfully recover the metric of the manifold.

Theorem 2 *Let $\mathcal{M}$ be a d-dimensional Riemannian manifold embedded in $\mathbb{R}^m$. If there exist a global isometry $\varphi : \mathcal{M} \to D \subset \mathbb{R}^d$, where D is an open connected subset of $\mathbb{R}^d$, then there is an orthonormal basis $\{V_i\}_{i=1}^d$ of the parallel fields on $\mathcal{M}$, and embedding function $f_i : \mathcal{M} \to \mathbb{R}$ whose gradient field satisfies $\nabla f_i = V_i, i = 1, \ldots, d$. Moreover, $F = (f_1, \ldots, f_d)$ is a global isometry.*

Proof In Euclidean space, if a vector field is written in Cartesian coordinates, then it is parallel if and only if it has constant coefficients (Petersen, 1998). Consider a parallel field on D. Since D is open connected, this parallel field is globally constant. Thus the space of parallel fields on D is a d dimensional linear space.

According to Proposition 5, we know that a global isometry preserves parallelism, that is, its differential carries a parallel vector field to another parallel vector field. Thus for a global isometry φ, $d\varphi$ maps parallel fields to parallel fields and $d\varphi$ is an isometric isomorphism, so is $d\varphi^{-1}$. Thus the space of parallel fields on $\mathcal{M}$ is isomorphic to the space of parallel fields on D. Therefore there exists an orthonormal basis $\{V_i\}_{i=1}^d$ of the space of the parallel fields on $\mathcal{M}$. Next we show V_i is

also a gradient field. We first map V_i to D using $d\varphi$. Clearly, $d\varphi(V_i)$ is a parallel field. Note that a parallel field in Euclidean space has constant coefficients in Cartesian coordinates. We can write $d\varphi(V_i)$ in Cartesian coordinates $\partial_j, j = 1,\ldots,d$ as follows:

$$d\varphi(V_i) = \sum_j c_i^j \partial_j,$$

where c_i^j are constant. Since $d\varphi$ is an isometric isomorphism, we can rewrite it as follows:

$$V_i = \sum_j c_i^j d\varphi^{-1}(\partial_j).$$

Here $d\varphi^{-1}(\partial_j)$ actually is an orthonormal basis of $\mathcal{M}$. Since c_i^j are constant, each V_i is a gradient field for some linear function with respect to the coordinates of $d\varphi^{-1}(\partial_j)$. Let f_i be a linear function such that $\nabla f_i = V_i$, $i = 1,\ldots,d$. It is worth noting that such a linear function f_i is not unique but only differs a constant. Then these $\{f_i | i = 1,\ldots,d\}$ constitutes a map F, $F = (f_1,\ldots,f_d) : \mathcal{M} \to \mathbb{R}^d$, which maps the manifold $\mathcal{M}$ to $\mathbb{R}^d$.

Next we show that such F is a global isometry on $\mathcal{M}$. Firstly, F is a local isometry. According to the construction of F, the differential of F $dF = (df_1,\ldots,df_d) = (V_1,\ldots,V_d)$ is orthonormal. Thus F is a local isometry according to Proposition 2. Secondly, F is a differmorphism. Clearly the map F restricted on the manifold $\mathcal{M}$, $F : \mathcal{M} \to F(\mathcal{M})$, is surjective. Next we show F is also injective. If not, assume there exist two distinct points p and q such that $F(p) = F(q)$. Then we have $f_i(p) = f_i(q)$, $i = 1,\ldots,d$. Since f_i is linear with respect to the coordinates of $d\varphi^{-1}(\partial_j)$. We rewrite f_i as follows:

$$f_i(p) = \sum_{j=1}^{d} c_i^j z_j(p) + \varepsilon_i,$$

where $z_j, j = 1,\ldots,d$ represent coordinate functions. Since c_i^j are constant, we have $z_j(p) = z_j(q)$ for $j = 1,\ldots,d$. Since z_j are coordinates, we have $p = q$ which contradicts to the assumption that p and q are distinct points. So far, we have proved F is a homeomorphism. Since each f_i is a linear function, F is clearly smooth. According to the Proposition 5.7 of Lee (2003), F^{-1} is also smooth, so F is a diffeomorphism. Since F is a local isometry and a diffeomorphism, it is a global isometry. ■

When there exists a global isometry, by optimizing our objective functions Equation (1), Equation (2) and Equation (4), Theorem 2 shows that our obtained gradient fields ∇f_i must be parallel. Moreover, the obtained map $F = (f_1,\ldots,f_d)$ is a global isometry. It might be worth noting that there are two variations in finding the global isometry. The first variation is the choice of the orthonormal basis of parallel fields. The second variation is the constant added to each embedding function. Thus the space of global isometry on $\mathcal{M}$ is actually $O(d) \times \mathbb{R}^d$. The first part is the space of orthonormal basis of parallel fields and the second part is the space of constants. Geometrically, the first part represents a rotation of the map and the second part represents a translation of the map.

When there is no isometry between the manifold $\mathcal{M}$ and $\mathbb{R}^d$, our approach can still find a reasonably good embedding function. For example, if the curvature of the manifold is not very high, by minimizing Equation (1), we can still find vector fields are nearly parallel. Consequently, the embedding would be nearly isometric. Please see our experimental results for details. However,

when the curvature of the manifold is extremely high, as shown in Figure 4, the obtained vector fields may have loops or singular points and is no longer a gradient field. The resulting embedding may cause overlap. It would be important to note that, in such cases, the isometric embedding or nearly isometric embedding does not exist.

3.3 Implementation

In real problems, the manifold $\mathcal{M}$ is usually unknown. In this subsection, we discuss how to find the isometric embedding function F from random points.

The implementation includes two steps, first we estimate parallel vector fields on manifold from random points, and then we reconstruct embedding functions by requiring that the gradient fields are as close to the parallel fields as possible. The parallel vector fields are computed one by one under orthogonality constraint. These two steps are described in subsections 3.3.1 and 3.3.2, respectively. After finding d orthonormal vector fields and the corresponding embedding function f_i, the final map F is given by $F = (f_1, \ldots, f_d)$. We discuss how to perform out-of-sample extension in subsection 3.3.3. The detailed algorithmic procedure is presented in subsection 3.3.4.

Given $x_i \in \mathcal{M} \subset \mathbb{R}^m$, $i = 1, ..., n$, we aim to find a lower dimensional Euclidean representation of the data such that the geometrical and topological properties can be preserved. We first construct a nearest neighbor graph by either ε-neighborhood or k nearest neighbors. Let $x_i \sim x_j$ denote that x_i is the neighbor of x_j or x_j is the neighbor of x_i. Let $N(i)$ denote the index set of the neighbors of x_i, that is, $N(i) = \{ j | x_j \sim x_i \}$. For each point x_i, we estimate its tangent space $T_{x_i}\mathcal{M}$ by performing principal component analysis on the neighborhood $N(i)$. We choose the largest d eigenvectors as the bases since $T_{x_i}\mathcal{M}$ is d dimensional. Let $T_i \in \mathbb{R}^{m \times d}$ be the matrix whose columns constitute a orthonormal basis for $T_{x_i}\mathcal{M}$. It is easy to show $P_i = T_i T_i^T$ is the *unique* orthogonal projection from $\mathbb{R}^m$ onto the tangent space $T_{x_i}\mathcal{M}$ (Golub and Loan, 1996). That is, for any vector $a \in \mathbb{R}^m$, we have $P_i a \in T_{x_i}\mathcal{M}$ and $(a - P_i a) \perp P_i a$.

3.3.1 PARALLEL VECTOR FIELD ESTIMATION

Let V be a vector field on manifold. For each point x_i, let V_{x_i} denote the value of the vector field V at x_i and $\nabla V|_{x_i}$ denote the value of ∇V at x_i. According to the definition of vector fields, V_{x_i} should be a tangent vector in the tangent space $T_{x_i}\mathcal{M}$. Thus it can be represented by local coordinates of the tangent space,

$$V_{x_i} = T_i v_i, \tag{5}$$

where $v_i \in \mathbb{R}^d$. We define $\mathbb{V} = \left(v_1^T, \ldots, v_n^T\right)^T \in \mathbb{R}^{dn}$. That is, $\mathbb{V}$ is a dn-dimensional big column vector which concatenates all the v_i's. By discretizing the objective function (1), the parallel field V can be obtained by solving the following optimization problem:

$$\begin{aligned} &\min_{\mathbb{V}} \; E(\mathbb{V}) = \sum_{i=1}^{n} \|\nabla V|_{x_i}\|_{\mathrm{HS}}^2, \\ &s.t. \;\; \|\mathbb{V}\| = 1. \end{aligned} \tag{6}$$

In the following we discuss, for a given point x_i, how to approximate $\|\nabla V|_{x_i}\|_{\mathrm{HS}}$.

Let $\gamma(t)$ be the geodesic connecting x_i and x_j which satisfies $\gamma(0) = x_i$ and $\gamma(d_{ij}) = x_j$, where d_{ij} is the geodesic distance of x_i and x_j. Let $e_{ij} = \gamma'(0)$. Since γ is a geodesic, $e_{ij} \in T_{x_i}\mathcal{M}$ is a unit

vector. Then the covariant derivative of vector field V along e_{ij} is given by (please see Figure 3)

$$\begin{aligned}
\nabla_{e_{ij}} V &= P_i\left(\frac{dV}{dt}|_{t=0}\right) \\
&= P_i \lim_{t\to 0} \frac{V(\gamma(t)) - V(\gamma(0))}{t} \\
&= P_i \frac{(V_{x_j} - V_{x_i})}{d_{ij}} \\
&\approx \sqrt{w_{ij}}(P_i V_{x_j} - V_{x_i}),
\end{aligned}$$

where $d_{ij} \approx 1/\sqrt{w_{ij}}$ approximates the geodesic distance d_{ij} of x_i and x_j. There are several ways to define the weights w_{ij}. Since for neighboring points, Euclidean distance is a good approximation to the geodesic distance, we can define the Euclidean weights as $w_{ij} = \frac{1}{\|x_i - x_j\|^2}$. If the data are uniformly sampled from the manifold, then w_{ij} would be almost constant. So in practice, $0-1$ weights is also widely used which is defined as follows:

$$w_{ij} = \begin{cases} 1, & \text{if } x_i \sim x_j \\ 0, & \text{otherwise.} \end{cases}$$

Since we do not know $\nabla_{\partial_i} V$ for a given basis ∂_i, $\|\nabla V\|^2_{\mathrm{HS}}$ can not be computed according to Equation (3). We define a $(0,2)$ symmetric tensor α as $\alpha(X,Y) = g(\nabla_X V, \nabla_Y V)$, where X and Y are vector fields on manifold. We have

$$\begin{aligned}
\mathrm{Trace}(\alpha) &= \sum_{i=1}^{d} g(\nabla_{\partial i} V, \nabla_{\partial_i} V) \\
&= \|\nabla V\|^2_{\mathrm{HS}},
\end{aligned}$$

where $\partial_1, \ldots, \partial_d$ is an orthonormal basis on tangent space. For the trace of α, we have the following geometric interpretation (see the exercise 1.12 in Chow et al., 2006):

$$\mathrm{Trace}(\alpha) = \frac{1}{\omega_d} \int_{S^{d-1}} \alpha(X,X) d\delta(X),$$

where S^{d-1} is the unit $(d-1)$-sphere, $d\omega_d$ is its volume, and $d\delta$ is its volume form. Thus for a given point x_i, we can approximate $\|\nabla V|_{x_i}\|_{\mathrm{HS}}$ by the following

$$\begin{aligned}
\|\nabla V|_{x_i}\|^2_{\mathrm{HS}} &= \mathrm{Trace}(\alpha)_{x_i} \\
&= \frac{1}{\omega_d} \int_{S^{d-1}} \alpha(X,X)|_{x_i} d\delta(X) \\
&\approx \sum_{j\in N(i)} \|\nabla_{e_{ij}} V\|^2 \\
&= \sum_{j\in N(i)} w_{ij} \left\|P_i V_{x_j} - V_{x_i}\right\|^2. \qquad (7)
\end{aligned}$$

For the third equation, the integral on the left hand side is approximated by the discrete summation on nearest neighbors. It might be worth noting that we implicitly assume that the nearest neighbors

are uniformly sampled. If the sampling is not uniform, then one should use weighted summation to approximate the integral.

Combining Equation (5), the optimization problem Equation (6) reduces to:

$$\begin{aligned}&\min_{\mathbb{V}} E(\mathbb{V}) = \sum_{i\sim j} w_{ij} \left\| P_i T_j v_j - T_i v_i \right\|^2, \\ &s.t.\ \|\mathbb{V}\| = 1.\end{aligned}$$

We first optimize $E(\mathbb{V})$ to find parallel vector fields on manifold, and then re-normalize the vector fields locally.

We will now switch to Lagrangian formulation of the problem. The Lagrangian is as follows:

$$\mathcal{L} = E(\mathbb{V}) - \lambda(\mathbb{V}^T\mathbb{V} - 1).$$

By matrix calculus, we have

$$\begin{aligned}&\frac{\partial E(\mathbb{V})}{\partial v_i} \\ =& -2\sum_{j\in N(i)} w_{ij}(T_i^T(P_iT_jv_j - T_iv_i) - T_i^TP_j(P_jT_iv_i - T_jv_j)) \\ =& 2\sum_{j\in N(i)} w_{ij}((T_i^TT_jT_j^TT_i + I_d)v_i - 2T_i^TT_jv_j) \\ =& 2\sum_{j\in N(i)} w_{ij}((Q_{ij}Q_{ij}^T + I_d)v_i - 2Q_{ij}v_j),\end{aligned}$$

where $Q_{ij} = T_i^TT_j$. Then we have

$$\frac{\partial E(\mathbb{V})}{\partial \mathbb{V}} = 2B\mathbb{V}, B = \begin{pmatrix} B_{11} & \cdots & B_{1n} \\ \vdots & \ddots & \vdots \\ B_{n1} & \cdots & B_{nn} \end{pmatrix},$$

where B is a $dn \times dn$ sparse block matrix. If we index each $d \times d$ block by B_{ij}, then for $i = 1, \ldots, n$, we have

$$B_{ii} = \sum_{j\in N(i)} w_{ij}(Q_{ij}Q_{ij}^T + I), \tag{8}$$

$$B_{ij} = \begin{cases} -2w_{ij}Q_{ij}, & \text{if } x_i \sim x_j \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

Requiring that the gradient of $\mathcal{L}$ vanish gives the following eigenvector problem:

$$B\mathbb{V} = \lambda\mathbb{V}.$$

In order to find multiple parallel fields, we just use the eigenvectors corresponding to the smallest eigenvalues of the matrix B. Recall Corollary 1 tells us that if we want to check the orthogonality of two parallel fields at every point, it suffices to compute the integral of the inner product of them

which can be discretely approximated as $\langle \mathbb{V}_i, \mathbb{V}_j \rangle$. Since the matrix B is symmetric, its eigenvectors are mutually orthogonal and, thus, $\langle \mathbb{V}_i, \mathbb{V}_j \rangle = 0$.

Recall Corollary 2 tells us for any point $p \in \mathcal{M}$, the tangent vector V_p of a parallel field V has a constant length. In our objective function (6), we only need to add a global normalization constraint $\|\mathbb{V}\| = 1$ for the sake of simplicity. After finding d vector fields $V_j, j = 1, \ldots, d$, we can further ensure local normalization as follows:

$$\|V_j|_{x_i}\| = 1, \forall i = 1, \ldots, n, j = 1, \ldots, d.$$

3.3.2 EMBEDDING

Once the parallel vector fields V_i are obtained, the embedding functions $f_i : \mathcal{M} \to \mathbb{R}$ can be constructed by requiring their gradient fields to be as close to V_i as possible. Recall that, if the manifold is isometric to Euclidean space, then the vector field computed via Equation (1) is also a gradient field. However, if the manifold is not isometric to Euclidean space, V may not be a gradient field. In this case, we try to find the optimal embedding function f in a least-square sense. This can be achieved by solving the following minimization problem:

$$\Phi(f) = \int_{\mathcal{M}} \|\nabla f - V\|^2 dx.$$

In order to discretize the above objective function, we first discuss the Taylor expansion of f on the manifold.

Let $\exp_x$ denote the exponential map at x. The exponential map $\exp_x : T_x\mathcal{M} \to \mathcal{M}$ maps the tangent space $T_x\mathcal{M}$ to the manifold $\mathcal{M}$. Let $a \in T_x\mathcal{M}$ be a tangent vector. Then there is a *unique* geodesic γ_a satisfying $\gamma_a(0) = x$ with initial tangent vector $\gamma_a'(0) = a$. The corresponding exponential map is defined by $\exp_x(ta) = \gamma_a(t)$, $t \in [0, 1]$. Locally, the exponential map is a diffeomorphism.

Note that $f \circ \exp_x : T_x\mathcal{M} \to \mathbb{R}$ is a smooth function on $T_x\mathcal{M}$. Then the following Taylor expansion of f holds:

$$f(\exp_x(a)) \approx f(x) + \langle \nabla f(x), a \rangle, \tag{10}$$

where $a \in T_x\mathcal{M}$ is a sufficiently small tangent vector. In discrete case, let $\exp_{x_i}$ denote the exponential map at x_i. Since $\exp_{x_i}$ is a diffeomorphism, there exists a tangent vector $a_{ij} \in T_{x_i}\mathcal{M}$ such that $\exp_{x_i}(a_{ij}) = x_j$. We approximate a_{ij} by projecting the vector $x_j - x_i$ to the tangent space, that is, $a_{ij} \approx P_i(x_j - x_i)$. Therefore, Equation (10) can be rewritten as follows:

$$f(x_j) = f(x_i) + \langle \nabla f(x_i), P_i(x_j - x_i) \rangle. \tag{11}$$

Since f is unknown, ∇f is also unknown. In the following, we discuss how to compute $\|\nabla f(x_i) - V_{x_i}\|$ discretely. We first show that the vector norm can be computed by an integral on a unit sphere, where the unit sphere can be discretely approximated by a neighborhood.

Let e be a unit vector on tangent space $T_x\mathcal{M}$, then we have (see the exercise 1.12 in Chow et al., 2006)

$$\frac{1}{\omega_d} \int_{S^{d-1}} \langle X, e \rangle^2 d\delta(X) = 1,$$

where S^{d-1} is the unit $(d-1)$-sphere, $d\omega_d$ its volume, and $d\delta$ its volume form. Let ∂_i, $i = 1, \ldots, d$, be an orthonormal basis on $T_x\mathcal{M}$. Then for any vector $b \in T_x\mathcal{M}$, it can be written as $b = \sum_{i=1}^{d} b^i \partial_i$.

Furthermore, we have

$$\begin{aligned}\|b\|^2 &= \sum_{i=1}^{d}(b^i)^2 \\ &= \sum_{i=1}^{d}(b^i)^2\frac{1}{\omega_d}\int_{S^{d-1}}\langle X,\partial_i\rangle^2 d\delta(X) \\ &= \frac{1}{\omega_d}\int_{S^{d-1}}\langle X,b\rangle^2 d\delta(X).\end{aligned}$$

From Equation (11), we see that

$$\langle \nabla f(x_i), P_i(x_j - x_i)\rangle = f(x_j) - f(x_i).$$

Thus, we have

$$\begin{aligned}&\|\nabla f(x_i) - V_{x_i}\|^2 \\ &= \frac{1}{\omega_d}\int_{S^{d-1}}\langle X, \nabla f(x_i) - V_{x_i}\rangle^2 d\delta(X) \\ &\approx \sum_{j\in N(i)}\langle e_{ij}, \nabla f(x_i) - V_{x_i}\rangle^2 \\ &= \sum_{j\in N(i)}\frac{1}{d_{ij}^2}\langle a_{ij}, \nabla f(x_i) - V_{x_i}\rangle^2 \\ &\approx \sum_{j\in N(i)} w_{ij}\langle P_i(x_j - x_i), \nabla f(x_i) - V_{x_i}\rangle^2 \\ &= \sum_{j\in N(i)} w_{ij}\left((P_i(x_j - x_i))^T V_{x_i} - f(x_j) + f(x_i)\right)^2,\end{aligned}$$

where e_{ij} is a unit vector and w_{ij} is the weight, and both of which are the same in Section 3.3.1. In the second equation, the integral is approximated by the discrete summation on nearest neighbors which is the same in Equation (7). In the fourth equation, the vector a_{ij} is approximated by the projection vector $P_i(x_j - x_i)$. Recall that In Section 3.3.1 d_{ij} is approximated by $\frac{1}{\sqrt{w_{ij}}}$ and we have $\|a_{ij}\| = d_{ij}$. Next we show these two approximations are coincide as long as $w_{ij} = O(\frac{1}{\|x_j - x_i\|^2})$. Let us take the Euclidean weight $w_{ij} = \frac{1}{\|x_j - x_i\|^2}$ as an example. We have

$$\lim_{x_j \to x_i}\frac{\|P_i(x_j - x_i)\|^2}{\|x_j - x_i\|^2} = \lim_{\theta\to 0}\cos(\theta)^2 = 1,$$

where θ is the angle between vector $P_i(x_j - x_i)$ and vector $x_j - x_i$.

Let $y_i = f(x_i)$ and $y = (y_1, \ldots, y_n)^T$. The objective function $\Phi(f)$ can be discretely approximated by $\Phi(y)$ as follows:

$$\Phi(y) = \sum_{i\sim j} w_{ij}\left((P_i(x_j - x_i))^T V_{x_i} - y_j + y_i\right)^2.$$

By setting $\partial\Psi(y)/\partial y = 0$, we get

$$-\sum_{i\sim j} w_{ij}s_{ij}(x_j - x_i)^T P_i V_{x_i} + \left(\sum_{i\sim j} w_{ij}s_{ij}s_{ij}^T\right) y = 0,$$

where s_{ij} is an all-zero vector except the i-th element being -1 and the j-th element being 1. Let $L = \sum_{i\sim j} w_{ij} s_{ij} s_{ij}^T$ and $c = \sum_{i\sim j} w_{ij} s_{ij} (x_j - x_i)^T P_i V_{x_i}$. Then we can rewrite the above linear system as follows

$$Ly = c. \tag{12}$$

Algorithm 1 PFE (Parallel Field Embedding)

Input: Data sample $X = (x_1, \ldots, x_n) \in \mathbb{R}^{m\times n}$
Output: $Y = (y_1, \ldots, y_n) \in \mathbb{R}^{d\times n}$
  for $i = 1$ to n **do**
    Compute tangent spaces $T_{x_i}M$
  end for
  Construct matrix B according to Equation (8), Equation (9)
  Find the smallest d eigenvalues $\lambda_1, \ldots, \lambda_d$ and the associated eigenvectors $\mathbb{V}_1, \ldots, \mathbb{V}_d$ of B
  for $l = 1$ to d **do**
    Construct vector field V_l from local representations $\mathbb{V}_l$
    Normalize $V_l(x_i)$ to unit-norm for each x_i
  end for
  Solve linear system $LY = c$
  return Y

It is easy to verify that L is a graph Laplacian matrix (Chung, 1997) and its rank is $n-1$. Thus, the solution of Equation (12) is not unique. If y^* is an optimal solution, $y^* + \text{constant}$ is also an optimal solution. By fixing $y_1 = 0$ in Equation (12), we get a unique solution of y. This is consistent with the continuous case. If f is an optimal solution of Equation (4), then $f + \text{const}$ is also an optimal solution.

3.3.3 OUT-OF-SAMPLE EXTENSION

Most nonlinear manifold learning algorithms do not have straightforward extension for out-of-sample examples. Some efforts (Bengio et al., 2003) are made to generalize existing nonlinear manifold learning algorithms to novel points. We will show that our PFE algorithm has a natural out-of-sample extension.

Given n training points $x_1, \ldots, x_n$ and n' new points $x_{n+1}, \ldots, x_{n+n'}$. Our task is to estimate the embedding results of the new points. For each new point x_j, we first find its k nearest points in the whole data set. Then we compute its tangent space $T_{x_j}\mathcal{M}$ by performing PCA on its neighborhood. We choose the largest d eigenvectors as the bases of $T_{x_j}\mathcal{M}$. Let $T_j \in \mathbb{R}^{m\times d}$ be the matrix whose columns constitute a orthonormal basis for $T_{x_j}\mathcal{M}$. For each vector V_{x_j}, it can be represented by local coordinates of the tangent space. That is, $V_{x_j} = T_j v_j$.

Since $\partial E/\partial \mathbb{V} = 2B\mathbb{V}$ and B is a symmetric matrix, we have

$$E(V) = \mathbb{V}^T B \mathbb{V}. \tag{13}$$

Let $\mathbb{V} = (\mathbb{V}_o^T, \mathbb{V}_n^T) = \left(v_1^T, \ldots, v_n^T, v_{n+1}^T, \ldots, v_{n+n'}\right)^T$, where $\mathbb{V}_o$ denotes the tangent vectors on the *original* training points and $\mathbb{V}_n$ denotes the tangent vectors on *new* points. Then Equation (13) can

be written as

$$E(\mathbb{V}) = \mathbb{V}^T B \mathbb{V} = \begin{bmatrix} \mathbb{V}_o^T & \mathbb{V}_n^T \end{bmatrix} \begin{bmatrix} B_{oo} & B_{on} \\ B_{no} & B_{nn} \end{bmatrix} \begin{bmatrix} \mathbb{V}_o \\ \mathbb{V}_n \end{bmatrix}.$$

Requiring $\partial E(\mathbb{V})/\partial \mathbb{V}_n = 0$, we obtain the following linear system:

$$B_{nn}\mathbb{V}_n = -B_{no}\mathbb{V}_o.$$

After solving this linear system, we get the optimal $\mathbb{V}_n$. Actually, one can compute f_n in the same way, where f_n denotes the function values on new points. We first construct the Laplacian matrix involving all points, including both new and old ones. Then taking derivatives with respect to f_n, we obtain the following linear system:

$$L_{nn}f_n = -L_{no}f_o,$$

where f_o denotes the function values on old points, and L_{nn} and L_{no} are block matrices which are defined in the same way as B_{nn} and B_{no}. Note that the procedure described above involves only local computation on each neighborhood of new samples and solving two sparse linear systems. Therefore our out-of-sample extension algorithm is quite efficient.

3.3.4 ALGORITHM

The PFE algorithm consists of three steps, which is summarized in Algorithm 1.

3.4 Related Work and Discussion

In this section, we would like to discuss the relationship between our work and related work which are based on Laplacian, Hessian and connection Laplacian operators.

The approximation of the Laplacian operator using the graph Laplacian Chung (1997) has enjoyed a great success in the last decade. Several theoretical results (Belkin and Niyogi, 2005; Hein et al., 2005) also showed the consistency of the approximation. One of the most important features of the graph Laplacian is that it is coordinate free. That is, the definition of the graph Laplacian does not depend on any special coordinate systems. The Laplacian operator based methods are motivated by the smoothness criterion, that is, the norm of the gradient $\int_{\mathcal{M}} \|\nabla f\|$ should be small. In the continuous case, with appropriate boundary conditions we have

$$\int_{\mathcal{M}} \|\nabla f\|^2 = \int_{\mathcal{M}} fL(f)dx. \tag{14}$$

Most of previous work focuses on approximating the continuous Laplacian operator. Next we show our method provides a direct way to approximate the integral on the left hand side of Equation (14). First note that

$$\|\nabla f\|^2 = \frac{1}{\omega_d} \int_{S^{d-1}} \langle X, b\rangle^2 d\delta(X).$$

From Equation (11), we see that

$$\langle \nabla f(x_i), P_i(x_j - x_i)\rangle = f(x_j) - f(x_i).$$

Therefore we have

$$\begin{aligned}\|\nabla f(x_i)\|^2 &= \frac{1}{\omega_d}\int_{S^{d-1}} \langle X, \nabla f(x_i)\rangle^2 d\delta(X) \\ &\approx \sum_{j\in N(i)} \langle e_{ij}, \nabla f(x_i)\rangle^2 \\ &= \sum_{j\in N(i)} \frac{1}{d_{ij}^2}\langle a_{ij}, \nabla f(x_i)\rangle^2 \\ &\approx \sum_{j\in N(i)} w_{ij}\langle P_i(x_j - x_i), \nabla f(x_i)\rangle^2 \\ &= \sum_{j\in N(i)} w_{ij}\left(f(x_i) - f(x_j)\right)^2.\end{aligned}$$

It can be seen that this result is consistent with traditional graph Laplacian methods.

Our method is also closely related to the approximation of Hessian operator. Note that if we replace V by ∇f in Equation (1), $E(V)$ becomes Hessian functional (Donoho and Grimes, 2003). It is evident by noticing that $\text{Hess} f = \nabla\nabla f$. The estimation of Hessian operator is very difficult and challenging. Previous approaches (Donoho and Grimes, 2003; Kim et al., 2009) first estimate normal coordinates on the tangent space, and then estimate the first order derivative of the function at each point, which turns out to be a matrix pseudo-inversion problem. One major limitation of this is that when the number of nearest neighbors k is larger than $d + \frac{d(d+1)}{2}$, where d is the dimension of the manifold, the estimation will be inaccurate and unstable (Kim et al., 2009). This is contradictory to the asymptotic case, since it is not desirable that k is bounded by a finite number when the data is sufficiently dense. In contrast, we directly estimate the norm of the second order derivative instead of trying to estimate its coefficients, which turns out to be an integral problem over the nearest neighbors. We only need to do simple matrix multiplications to approximate the integral at each point, but do not have to solve matrix inversion problems. Therefore, asymptotically, we would expect our method to be much more accurate and robust for the approximation of the norm of the second order derivative.

The most related work is the Vector Diffusion Maps (VDM, Singer and Wu, 2011) as we both focus on vector fields rather than embedding functions. VDM is based on the heat kernel for vector fields rather than for functions over the data. It first constructs the heat kernel and orthogonal transformations from the weighted graph. Then VDM defines an embedding of the data via full spectral decomposition of the heat kernel. VDM tries to preserve the newly defined vector diffusion distance for the data when doing embedding. Singer and Wu (2011) also provided theoretical analysis that the construction for the kernel essentially defines the discrete type of the *connection Laplacian operator* and they proved the consistency of such an approximation. We first show that the objective function of finding parallel vector fields of PFE is the same as VDM. According to the Bochner technique (please see Section 3.2 in Chapter 7 of Petersen, 1998), with appropriate boundary conditions we have

$$\int_{\mathcal{M}} \|\nabla V\|_{\text{HS}}^2 = \int_{\mathcal{M}} \langle \nabla^*\nabla V, V\rangle, \tag{15}$$

where $\nabla^*\nabla$ is the connection Laplacian operator. VDM approximated the connection Laplacian operator by generalizing the graph Laplacian operator. We propose to directly approximate the integral on the left hand side of Equation (15). The approximations of PFE and VDM share several

similar important features but also differ in several aspects. Firstly, we use the same way to represent vector fields by using local coordinates of tangent spaces. Intuitively, this is the most natural way to represent vector fields as long as the data are embedded in Euclidean space, or in other words the data has features. It would be very interesting to consider the problem of how to represent vector fields on the graph data, that is, the data that do not have features. Secondly, the approximation of the covariant derivative is similar but different. The idea of computing the covariant derivative is to find a way to compute the difference between vectors on different tangent spaces. VDM proposed an intrinsic way to compute covariant derivative using the concept of *parallel transport*. They first transported the vectors to the same tangent space using the parallel transport, and then compute the difference of vectors on the tangent space. The way of finding the parallel transport between two points is to compute the orthogonal transformation between two corresponding tangent spaces. It turns out that computing the parallel transport is a singular value decomposition problem for each edge of the nearest neighbor graph. Our approach first computes the directional derivative using the (parallel) transport of vectors on Euclidean space, then projects the directional derivative to the corresponding tangent space. The main computational cost is the projection which is the multiplication of matrices with vectors. In continuous cases, they are two different ways to define the covariant derivative. Thirdly, the discrete type connection Laplacian operators (matrix $D^{-1}S-I$ in VDM and matrix B in PFE) are different. The difference is that the transformation matrix O_{ij} in VDM is orthogonal but the transformation matrix Q_{ij} in PFE is not. It is because that we use different ways to approximate the covariant derivative. It might be also worth noting that if we use the orthogonal transformation matrix to compute the covariant derivative, the resulted connection Laplacian matrix followed by our discrete approximation methods would be the same as VDM. Overall, VDM uses vector fields to define the vector diffusion distance, while PFE uses vector fields to find the isometry. Although the procedures of computing vector fields are similar, the motivation and objective are different.

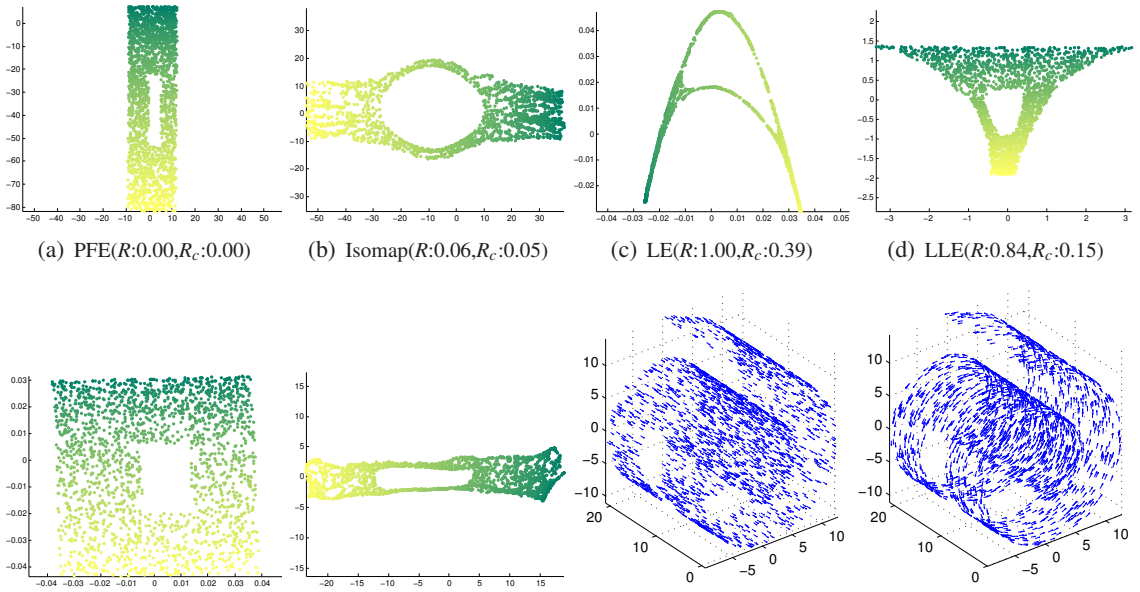
4. Experiments

In this section, we evaluate our algorithm on several synthetic manifold examples and two real data sets.

4.1 Topology

In this example, we study the effectiveness of different manifold learning algorithms for isometric embedding. The data set contains 2000 points sampled from a swiss roll with a hole, which is a 2D manifold embedded in $\mathbb{R}^3$. The swiss roll is a highly nonlinear manifold. Classical linear algorithms like PCA cannot preserve the manifold structure. On the other hand, the swiss roll is a *flat* manifold with zero-curvature everywhere and thus can be isometrically embedded in $\mathbb{R}^2$. We compare our algorithm with several state-of-the-art nonlinear dimensionality reduction algorithms: Isomap, Laplacian Eigenmaps (LE), Locally Linear Embedding (LLE), Hessian Eigenmaps (HLLE), and Maximum Variance Unfolding (MVU).

In all of our experiments, we use a binary-weighted k nearest neighbor graph for each algorithm that needs to construct a graph. Since the manifold learning algorithms usually rely heavily on the choice of the number of nearest neighbors for graph construction, we run each algorithm with the number of nearest neighbors (k) varying in $\{4, 5, \ldots, 20\}$ and show the best results of each algorithm according to the R-score (to be introduced shortly).



(a) PFE(R:0.00,R_c:0.00) (b) Isomap(R:0.06,R_c:0.05) (c) LE(R:1.00,R_c:0.39) (d) LLE(R:0.84,R_c:0.15)

(e) HLLE(R:1.00, R_c:0.21) (f) MVU(R:0.38,R_c:0.11) (g) V_1 (h) V_2

Figure 5: Isometric embedding of 2000 points on a Swiss Roll with a hole. The number of nearest neighbors (k) is set to the best among $\{4, 5, \ldots, 20\}$ for each algorithm when constructing the neighborhood graph. (a)-(f) show the embedding results of various algorithms. (g)-(h) visualize the two vector fields obtained by our algorithm.

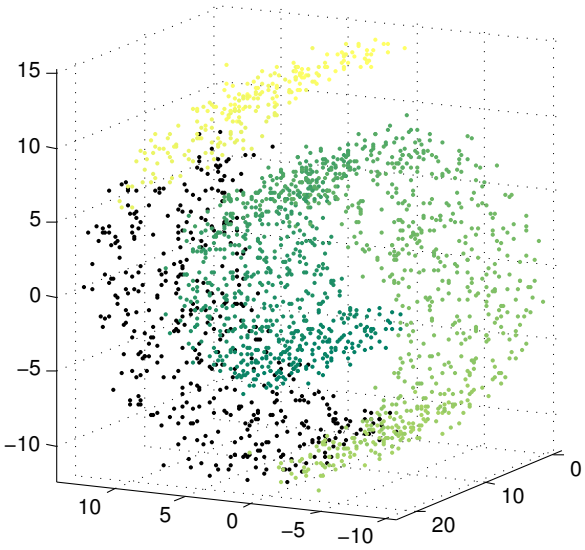


Figure 6: 2000 points are randomly sampled from the Swiss roll with noise. The black points are used for visualization of the gradient field.

The embedding results for all algorithms are shown in Figure 5. Both LE and LLE fail to recover the intrinsic rectangular shape of the original manifold. Isomap performs well at the two ends of the swiss roll but generate a big distorted hole in the middle. This is due to the fact that Isomap can not handle non-convex data. MVU also roughly preserved the overall rectangular shape, but it still generates distortions around the hole. Both HLLE and PFE give very good results. However, it would be important to note that our PFE algorithm generates an *isometric* embedding while the result of HLLE fail to preserve the scale of the coordinates. In order to measure the faithfulness of the embedding in a quantitative way, we employ the normalized R-score (Goldberg and Ritov, 2009):

$$R(X,Y) = \frac{1}{n}\sum_{i=1}^{n} G(X_i,Y_i)/\|HX_i\|_F^2.$$

Here $G(X_i,Y_i)$ is the (normalized) *Procrustes statistic* (Sibson, 1978) which measures the distance between two configurations of points (the original data X_i and the embedded Y_i). $H = I - \frac{1}{k}\mathbf{1}\mathbf{1}^T$ is the centering matrix. A local isometry preserving embedding is considered faithful and thus would get a low R-score. We also report a variant R_c-score (Goldberg and Ritov, 2009)

$$R_c(X,Y) = \frac{1}{n}\sum_{i=1}^{n} G_c(X_i,Y_i)/\|HX_i\|_F^2,$$

where $G_c(X_i,Y_i)$ allows not only rotation and translation, but also rescaling when measuring the distance between two configurations of points. This score can be used to measure conformal map. Please refer to Goldberg and Ritov (2009) for the details of R-score and R_c-score.

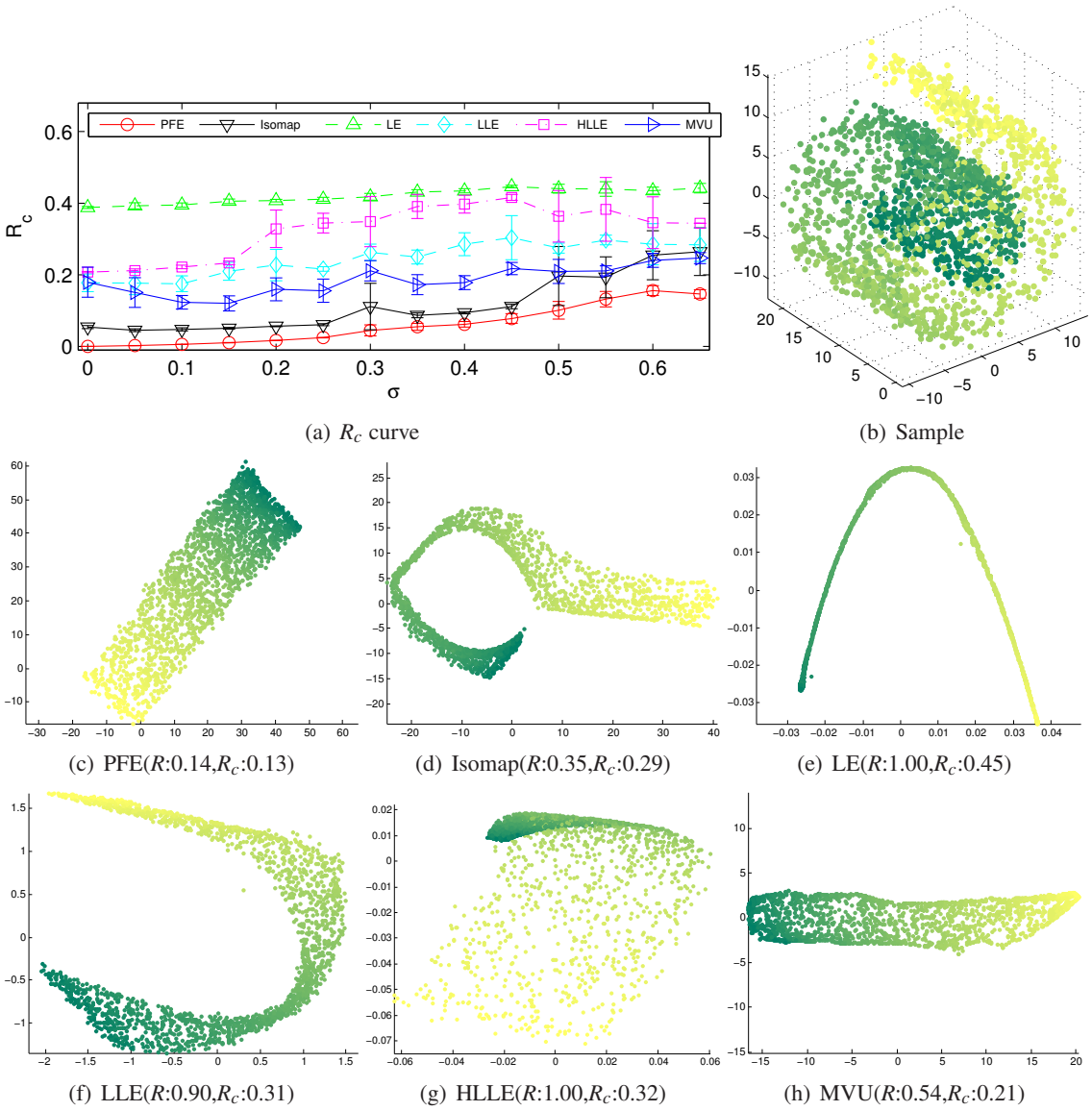
We give the measures of R-score and R_c-score for each algorithm in Figure 5. As can be seen, PFE outperforms all the other algorithms by achieving the minimum R-score and R_c-score. For R-score, except for PFE, Isomap and MVU, all the other three algorithms give excessively high value because their normalization significantly change the scale of the original coordinates. These three algorithm also performs worse than MVU, Isomap and especially PFE in terms of R_c-score.

4.2 Noise

In this example, we compare the performance of different algorithms on noisy data. 2000 points are randomly sampled from the swiss roll without hole, as shown in Fig 6. Then we add random Gaussian noises $\mathcal{N}(0,\sigma^2)$ to each dimension. For each given σ^2, we repeat our experiment 10 times with random noise and the average and standard deviation of the R_c-scores are recorded for each algorithm. The results are shown in Figure 7(a). As can be seen, our PFE method consistently outperforms other algorithms and is relatively stable even under heavy noise. Isomap performs the second best. It also achieves very small standard deviation under small noises, but it becomes very unstable when $\sigma \geq 0.5$.

Figure 7(b) shows the sample points when $\sigma = 0.65$ and Figure 7(c)$\sim$(h) show the embedding results of different algorithms. As can be seen, under such heavy noise, Isomap, LE, and LLE distort the original Swiss roll. HLLE can expand the manifold correctly to some extent, but there is overlap at the top. MVU unfolds the manifold correctly, but does not preserve the isometry well. Our PFE algorithm can successfully recover the intrinsic structure of the manifold.

Note that, algorithms other than PFE do not calculate vector fields explicitly. However, once the embedding result is obtained, we can reversely estimate the gradient field at each point x_i, which is



(a) R_c curve (b) Sample

(c) PFE(R:0.14,R_c:0.13) (d) Isomap(R:0.35,R_c:0.29) (e) LE(R:1.00,R_c:0.45)

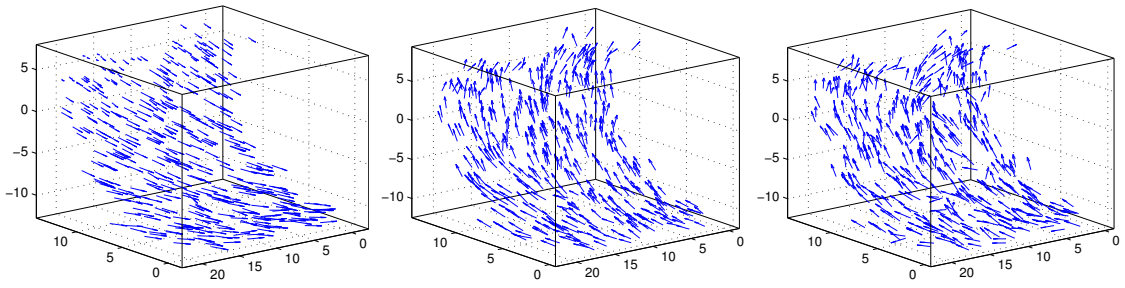
(f) LLE(R:0.90,R_c:0.31) (g) HLLE(R:1.00,R_c:0.32) (h) MVU(R:0.54,R_c:0.21)

Figure 7: Swiss roll with noise. (a) shows the R_c-scores of the five algorithms. (b) shows the 2000 random points sampled from the swiss roll with noise ($\sigma = 0.65$). (c∼h) show the embedding results of the six algorithms.

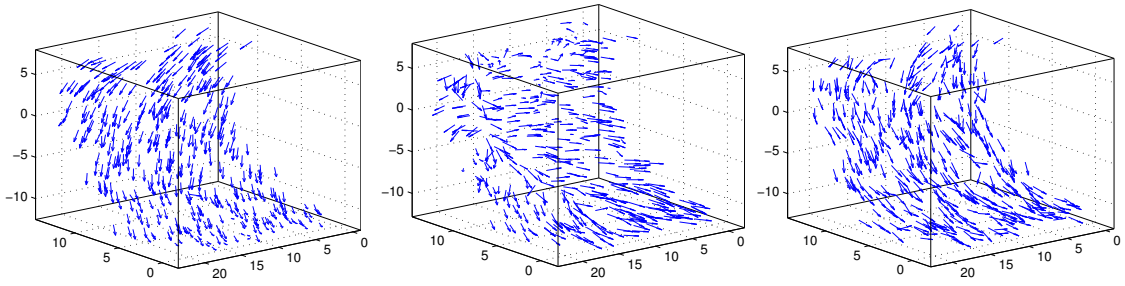
denoted as $\nabla f(x_i)$, by minimizing the following objective function at the local neighborhood of x_i :

$$\sum_{j\sim i}(f(x_j) - f(x_i) - (P_i(x_j - x_i)) \cdot \nabla f(x_i))^2.$$

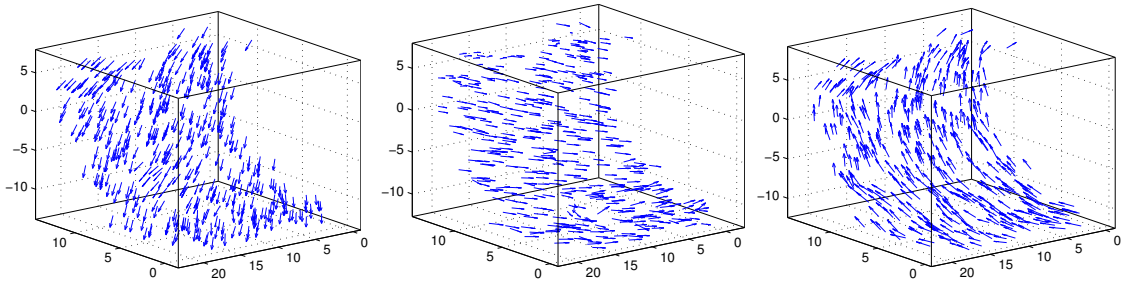
Figure 8 shows the gradient fields obtained by various algorithms. For the purpose of better visualization, we only show the gradient field at part of the Swiss roll (i.e., the black points in Figure 6).



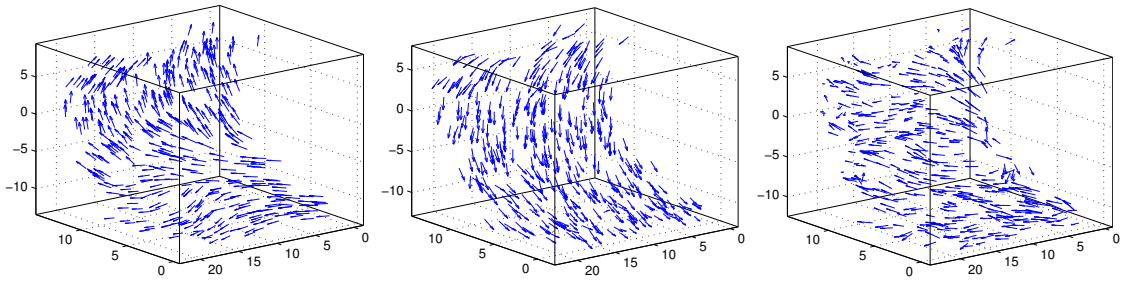
(a) PFE-1 (b) Isomap-1 (c) LE-1



(d) PFE-2 (e) Isomap-2 (f) LE-2

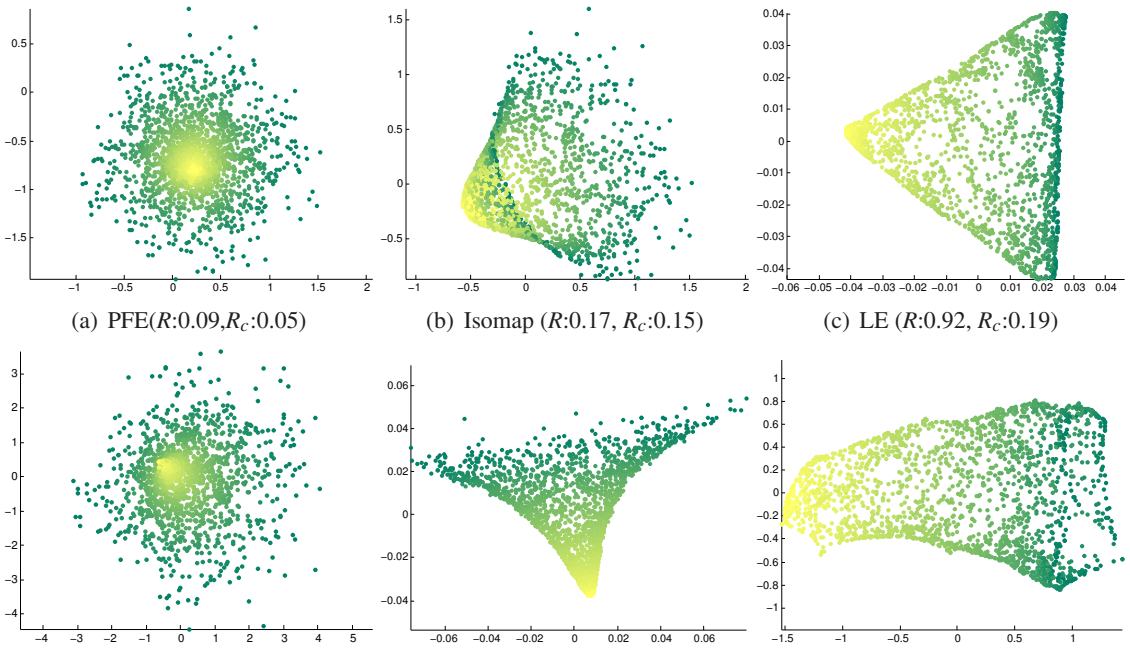


(g) LLE-1 (h) HLLE-1 (i) MVU-1



(j) LLE-2 (k) HLLE-2 (l) MVU-2

Figure 8: The gradient fields obtained by the six algorithms for the Swiss roll. For the sake of better visualization, only part of the gradient fields is shown.



(a) PFE(R:0.09,R_c:0.05) (b) Isomap (R:0.17, R_c:0.15) (c) LE (R:0.92, R_c:0.19)

(d) LLE (R:1.79, R_c:0.08) (e) HLLE (R:0.92, R_c:0.21) (f) MVU (R:0.58, R_c:0.11)

Figure 9: Embedding results of Gaussian. (a∼f) show the embedding results of the six algorithms.

As can be seen, even with noise, our PFE algorithm can accurately find two orthogonal smooth vector fields.

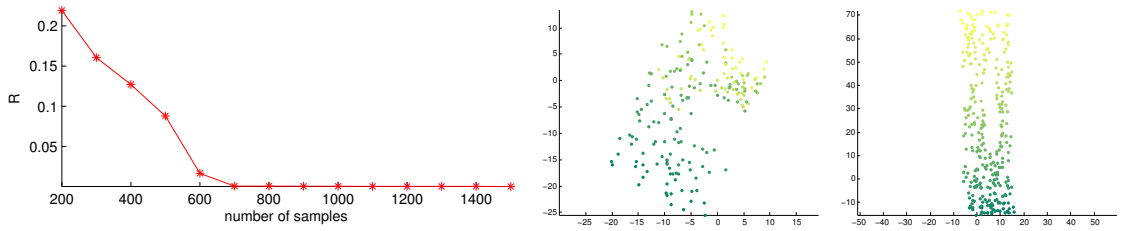
4.3 Curvature

Our PFE algorithm tries to find an isometric embedding. When the underlying manifold cannot be isometrically embedded in $\mathbb{R}^d$, PFE seeks for a least square approximation. The data set used in this experiment is a Gaussian surface. Note that, there is no isometric map from this surface to $\mathbb{R}^2$. The embedding results of the six algorithms are shown in Figure 9. As can be seen, except PFE and LLE, for all the other four algorithms, there is big distortion or overlap. LLE still produces small distortions at the center of the embedding. Moreover, PFE has the advantage over LLE that it optimally preserves the global isometry, and therefore achieving much smaller R-score. Even when measured by the R_c score, PFE outperforms LLE and all the other algorithms.

4.4 Sampling Rate

In this subsection, we test the robustness of our algorithm with respect to the density of data points. Specifically, we use the swiss roll with a hole manifold, and test our algorithm under different sampling rates. As shown in Figure 10(a), we run our algorithm on a data set with the number of data points varying from 200 to 1500 and report the corresponding R-score.

As can be seen, our algorithm produces near-optimal results when the number of points reaches 700. The R-score keeps increasing when the sampling rate reduces. This agrees with the common sense that it is difficult to analyze a small number of samples. However, the R-score does not increase too much even with small sampling rate. As we visualized in Figure 10(b) and (c),



(a) R-score curve (b) 200 points (c) 300 points

Figure 10: Robustness with respect to the sampling rate. (a) shows the R-score achieved by PFE on the swiss roll with hole data set, with respect to different numbers of samples (i.e., different sampling rates). (b) and (c) give the visualization of the embedding obtained from 200 samples and 300 samples, respectively.

although the result of 200 points is distorted and overlapped, the result of 300 points is good. This demonstrates the robustness of our algorithm with respect to different sampling rates.

4.5 Face Manifold

We consider the application of our algorithm on a real-world face data set. The data set[2] contains 698 64×64 gray-scale face images. So each face image is represented as a point in the 4096-dimensional Euclidean space. However, the intrinsic dimensionality may be very low, because the face images are rendered with variations on two pose parameters (up-down and left-right) and one lighting condition parameter. By applying our PFE algorithm, the face images are embedded in a two-dimensional space as shown in Figure 11. We show some representative images selected along the boundary of the embedding. As can be seen, the new coordinates can accurately reflect the face variations.

4.6 Classification after Embedding

In many scenarios, calculating the low-dimensional embedding of the data manifold is not the final step. So another popular evaluation of dimensionality reduction algorithms is to measure the performance of general machine learning algorithms on the low-dimensional representation. In this subsection, we compare the dimensionality reduction algorithms by evaluating the accuracy of 1NN classification on the low-dimensional data.

Specifically, we work with the CMU PIE face data set (Sim et al., 2003). This data set contains 68 subjects with 41,368 face images as a whole. The face images were captured by 13 synchronized cameras and 21 flashes, under varying pose, illumination and expression. We use the frontal pose (C27) subset with varying lighting and illumination in this experiment. Each face image is of the size 32×32, and we simply work with the raw pixel features without preprocessing. For each algorithm, we calculate the low-dimensional embedding, and then run face recognition with simple Nearest-neighbor classifier. The classification results for 2, 3, 4 and 5 training labels for each person[3] are

2. It is available online from `http://isomap.stanford.edu/face_data.mat.Z`.
3. Labels are randomly chosen and 10 repetitions are run to calculate the averaged performance.

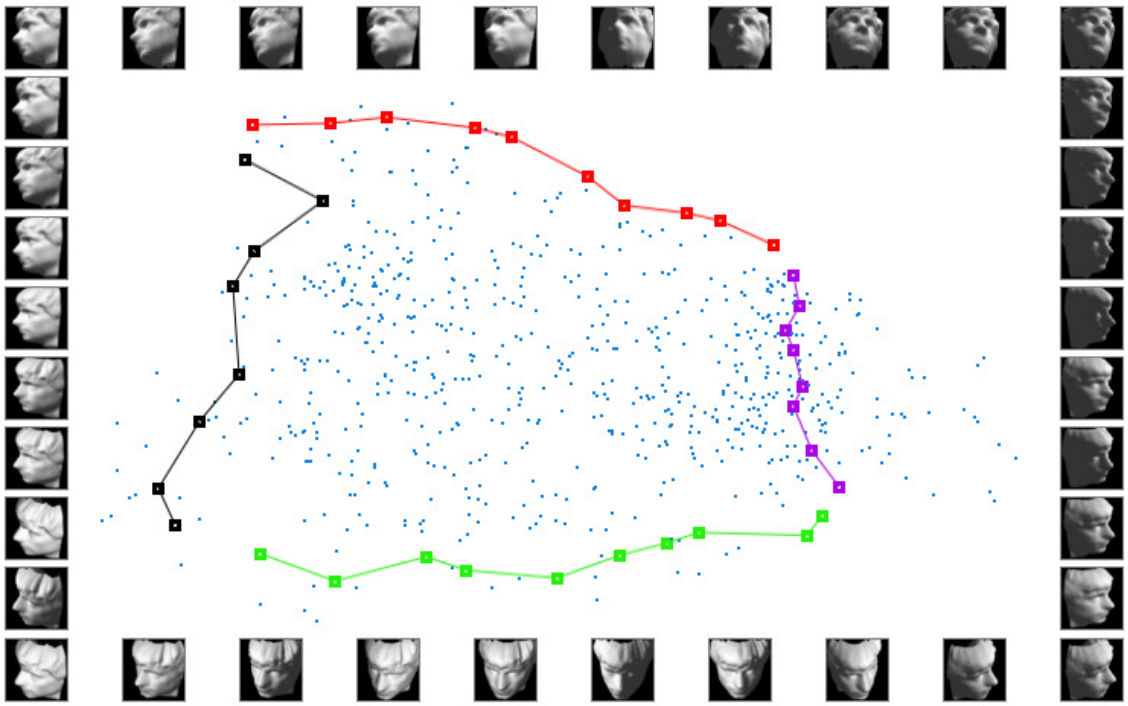


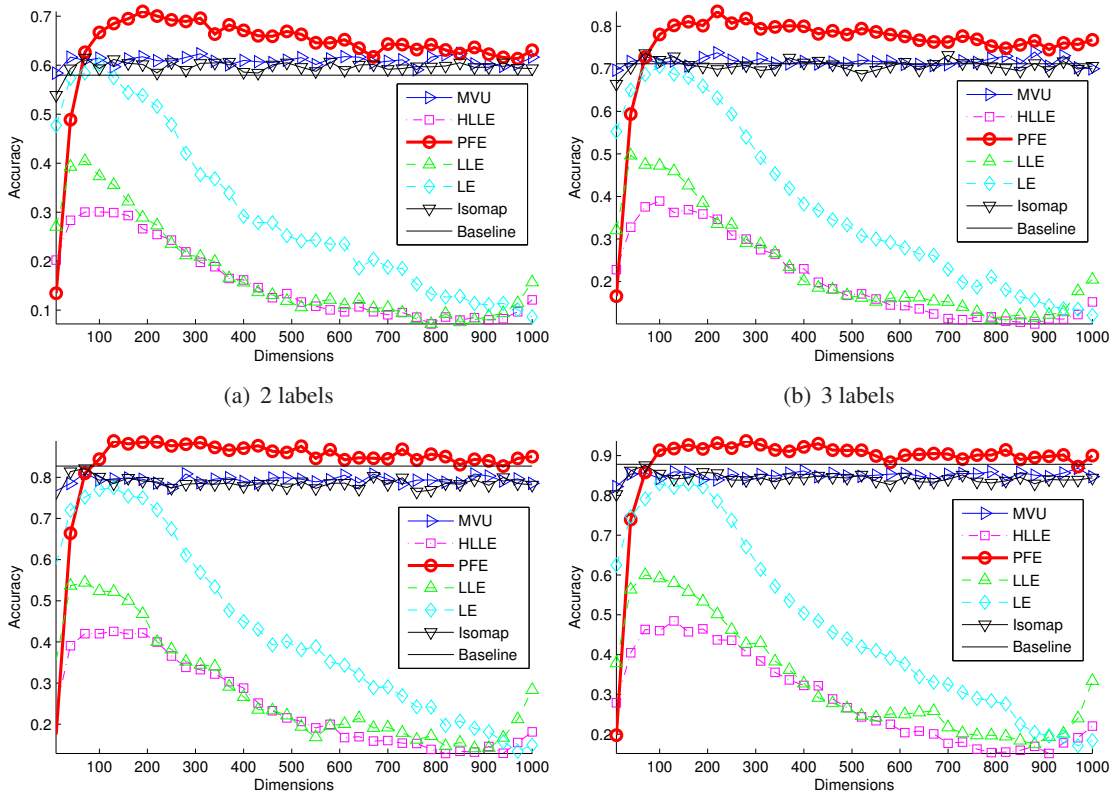
Figure 11: Face manifold embedding. A face data set of 698 images is embedded in the 2-dimensional Euclidean space. Some representative face images are selected along the boundary.

shown in Figure 12. We also show the baseline performance, which corresponds to classification in the original feature space without dimensionality reduction.

As we can see, PFE performs best around dimension 250. It is interesting to see that PFE does not outperform other algorithm when the dimension is very low. But a significant performance gap over other algorithms as well as the baseline can be observed when comparing the peak performance.

4.7 Out-of-Sample Extension

In this experiment, we evaluate the effectiveness of our PFE algorithm for out-of-sample extension. The data set used here is the face data set used in Section 4.5. Specifically, we leave 4 points (corresponding to 4 different poses) out and compute the embedding of the remaining 694 points. The embedding result is shown in Figure 13, which looks almost identical to the previous result in Figure 11. Then we compute the embedding of the 4 testing points by using our out-of-sample extension algorithm. The 4 testing points associated with their face images are shown in Figure 13. As can be seen, these testing samples optimally find their coordinates which reflect their intrinsic property, that is, pose variation. Note beside poses, there is one extra degree of freedom (light condition) in this data set. By embedding the data points in a plane, we ignore the factor of light condition.



(a) 2 labels (b) 3 labels

(c) 4 labels (d) 5 labels

Figure 12: Classification after embedding. Baseline shows the performance without dimensionality reduction.

We also demonstrate the performance of out-of-sample extension by using synthetic data. Specifically, 300 points are uniformly sampled from the swiss roll and embedded in a two-dimensional space by using our PFE algorithm. Then we randomly sample 5000 other points from the same manifold and embed them by using our out-of-sample extension algorithm. The 300 training points and the final embedding results of all samples are shown in Figure 14(a) and Figure 14(b), respectively. As can be seen, our algorithm can find a faithful embedding for both training and testing points.

5. Conclusion

We have introduced a novel local isometry based dimensionality reduction method from the perspective of vector field. Learning on manifold has three most important aspects: geometry, topology and functions on the manifold. Interestingly, there is strong connection between these three aspects and the vector fields on the manifold. In this paper, we are particularly interested in the connection between the geometry and the vector fields on the manifold. A variational method is proposed to find vector fields as parallel as possible on the manifold. The property of such vector field reflects the in-

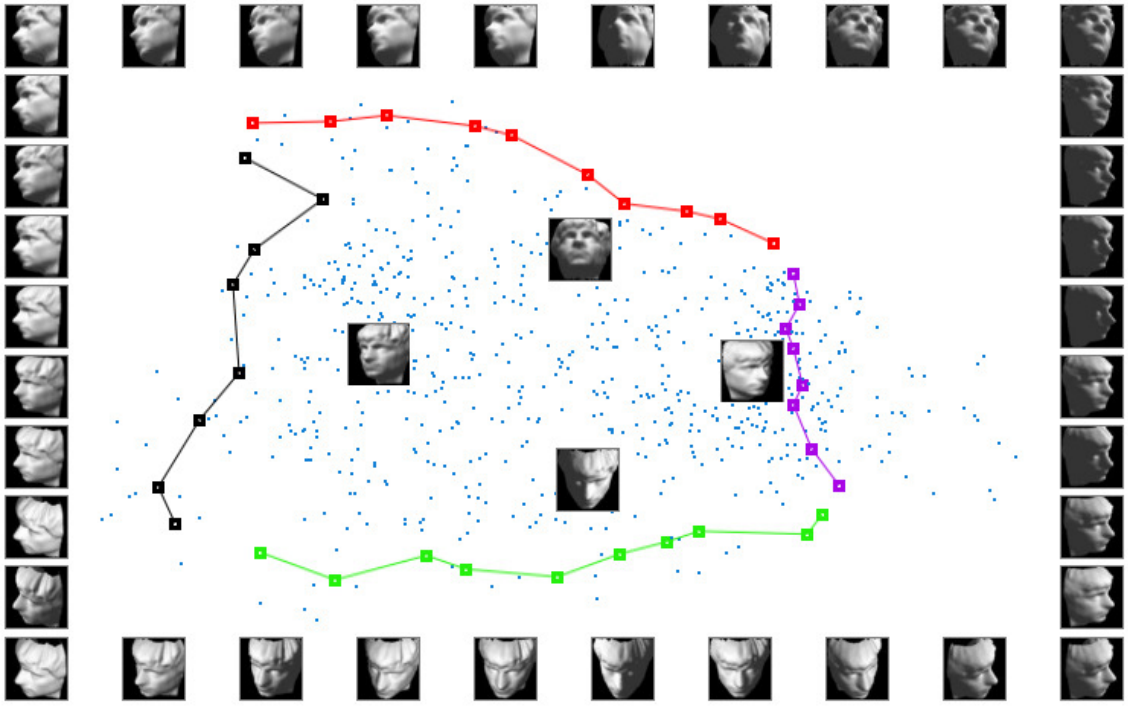
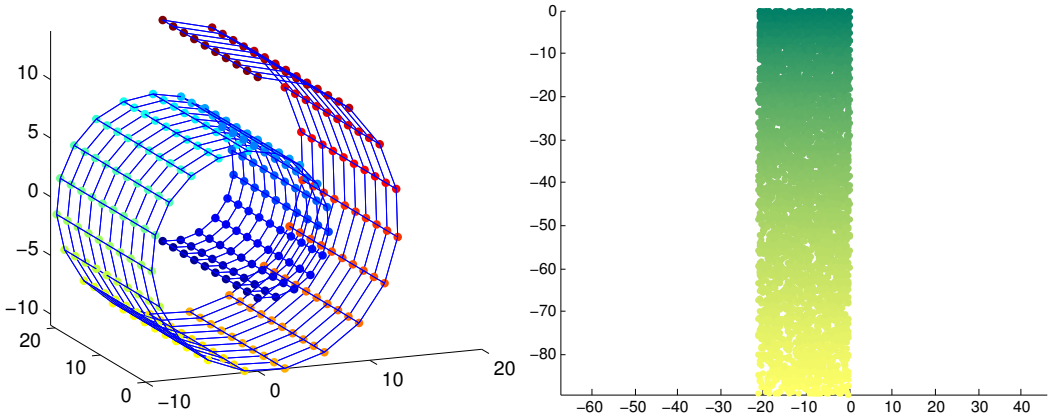


Figure 13: Out-of-sample extension for the face data set. As can be seen, these four testing samples optimally find their coordinates which reflect their intrinsic property, that is, pose variation.



(a) Training Samples (b) Embedding

Figure 14: Out-of-sample extension on the Swiss Roll. (a) shows the 300 uniformly sampled training points. (b) shows the embedding results of both the 300 training points and the 5000 testing points. Both R-score and R_c-score equal to 0.00 in this experiment.

trinsic geometry and topology of the manifold. If the manifold is isometric to Euclidean space, then the obtained vector field is parallel. If the manifold has high curvature or complex topology, then the obtained vector field may be twisted and may have loops or singular points. These properties can be used to study the intrinsic structure of the manifold.

Parallel fields play a central role for finding an isometry. Moreover, parallel fields provide a natural parametric representation of the manifold. Besides dimensionality reduction, they are also useful for other learning problems on the manifold. For example, we can perform classification and regression on the manifold by requiring that the function varies smoothly along the vector fields.

Acknowledgments

This work is supported by National Basic Research Program of China (973 Program) (Grant No: 2012CB316400) and National Natural Science Foundation of China (Grant No: 61125203, 61233011, 90920303).

References

M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*, pages 585–591. MIT Press, Cambridge, MA, 2001.

M. Belkin and P. Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. In *COLT*, pages 486–500, 2005.

M. Belkin and P. Niyogi. Convergence of laplacian eigenmaps. In *Advances in Neural Information Processing Systems 19*, pages 129–136. 2007.

Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems 16*, 2003.

M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 16*, 2003.

B. Chow, P. Lu, and L. Ni. *Hamilton's Ricci Flow*. AMS, Providence, Rhode Island, 2006.

F. R. K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. AMS, 1997.

R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21 (1):5 – 30, 2006. Diffusion Maps and Wavelets.

T. Cox and M. Cox. *Multidimensional Scalling*. Chapman & Hall, London, 1994.

A. Defant and K. Floret. *Tensor Norms and Operator Ideals*. North-Holland Mathematics Studies, North-Holland, Amsterdam, 1993.

P. Dollár, V. Rabaud, and S. Belongie. Non-isometric manifold learning: Analysis and an algorithm. In *Proceedings of the Twenty-fourth International Conference on Machine Learning*, pages 241–248, 2007.

D. L. Donoho and C. E. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences of the United States of America*, 100(10):5591–5596, 2003.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, Hoboken, NJ, 2nd edition, 2000.

Y. Goldberg and Y. Ritov. Local procrustes for manifold embedding: a measure of embedding quality and embedding algorithms. *Machine Learning*, 77(1):1–25, 2009.

Y. Goldberg, A. Zakai, D. Kushnir, and Y. Ritov. Manifold learning: The price of normalization. *The Journal of Machine Learning Research*, 9:1909–1939, 2008.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

J. Ham, D. D. Lee, Sebastian M., and Bernhard S. A kernel view of the dimensionality reduction of manifolds. In *Proceedings of the Twenty-first International Conference on Machine Learning*, 2004.

M. Hein, J. Audibert, and U. V. Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph laplacians. In *COLT*, pages 470–485. Springer, 2005.

I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1989.

K. I. Kim, F. Steinke, and M. Hein. Semi-supervised regression using hessian energy with an application to semi-supervised dimensionality reduction. In *Advances in Neural Information Processing Systems 22*, pages 979–987. 2009.

S. Lafon and A. B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1393–1403, 2006.

J. M. Lee. *Introduction to Smooth Manifolds*. Springer Verlag, New York, 2nd edition, 2003.

T. Lin and H. Zha. Riemannian manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):796–809, 2008.

B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In *Advances in Neural Information Processing Systems 18*, pages 955–962. 2006.

P. Petersen. *Riemannian Geometry*. Springer, New York, 1998.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

B. Schölkopf, A. Smola, and K. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, (10):1299–1319, 1998.

R. Sibson. Studies in the robustness of multidimensional scaling: Procrustes statistics. *Journal of the Royal Statistical Society. Series B (Methodological)*, 40(2):234–238, 1978.

T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2003.

A. Singer and H.-t. Wu. Vector diffusion maps and the connection Laplacian. *ArXiv e-prints*, 2011.

J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the Twenty-first International Conference on Machine Learning*, 2004.

Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM Journal of Scientific Computing*, 26(1), 2004.

Journal of Machine Learning Research 14 (2013) 2979-3010 Submitted 10/12; Revised 6/13; Published 10/13

Multi-Stage Multi-Task Feature Learning

Pinghua Gong GPH08@MAILS.TSINGHUA.EDU.CN
State Key Laboratory on Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Automation, Tsinghua University
Beijing 100084, China

Jieping Ye JIEPING.YE@ASU.EDU
Computer Science and Engineering
Center for Evolutionary Medicine and Informatics
The Biodesign Institute
Arizona State University
Tempe, AZ 85287, USA

Changshui Zhang ZCS@MAIL.TSINGHUA.EDU.CN
State Key Laboratory on Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Automation, Tsinghua University
Beijing 100084, China

Editor: Hui Zou

Abstract

Multi-task sparse feature learning aims to improve the generalization performance by exploiting the shared features among tasks. It has been successfully applied to many applications including computer vision and biomedical informatics. Most of the existing multi-task sparse feature learning algorithms are formulated as a convex sparse regularization problem, which is usually suboptimal, due to its looseness for approximating an ℓ_0-type regularizer. In this paper, we propose a non-convex formulation for multi-task sparse feature learning based on a novel non-convex regularizer. To solve the non-convex optimization problem, we propose a Multi-Stage Multi-Task Feature Learning (MSMTFL) algorithm; we also provide intuitive interpretations, detailed convergence and reproducibility analysis for the proposed algorithm. Moreover, we present a detailed theoretical analysis showing that MSMTFL achieves a better parameter estimation error bound than the convex formulation. Empirical studies on both synthetic and real-world data sets demonstrate the effectiveness of MSMTFL in comparison with the state of the art multi-task sparse feature learning algorithms.

Keywords: multi-task learning, multi-stage, non-convex, sparse learning

1. Introduction

Multi-task learning (MTL) (Caruana, 1997) exploits the relationships among multiple related tasks to improve the generalization performance. It has been successfully applied to many applications such as speech classification (Parameswaran and Weinberger, 2010), handwritten character recognition (Obozinski et al., 2006; Quadrianto et al., 2010) and medical diagnosis (Bi et al., 2008). One common assumption in multi-task learning is that all tasks should share some common structures

including the prior or parameters of Bayesian models (Schwaighofer et al., 2005; Yu et al., 2005; Zhang et al., 2006), a similarity metric matrix (Parameswaran and Weinberger, 2010), a classification weight vector (Evgeniou and Pontil, 2004), a low rank subspace (Chen et al., 2010; Negahban and Wainwright, 2011) and a common set of shared features (Argyriou et al., 2008; Gong et al., 2012; Kim and Xing, 2009; Kolar et al., 2011; Lounici et al., 2009; Liu et al., 2009; Negahban and Wainwright, 2008; Obozinski et al., 2006; Yang et al., 2009; Zhang et al., 2010).

Multi-task feature learning, which aims to learn a common set of shared features, has received a lot of interests in machine learning recently, due to the popularity of various sparse learning formulations and their successful applications in many problems. In this paper, we focus on a specific multi-task feature learning setting, in which we learn the features specific to each task as well as the common features shared among tasks. Although many multi-task feature learning algorithms have been proposed in the past, many of them require the relevant features to be shared by all tasks. This is too restrictive in real-world applications (Jalali et al., 2010). To overcome this limitation, Jalali et al. (2010) proposed an $\ell_1 + \ell_{1,\infty}$ regularized formulation, called "dirty model", to leverage the common features shared among tasks. The dirty model allows a certain feature to be shared by some tasks but not all tasks. Jalali et al. (2010) also presented a theoretical analysis under the incoherence condition (Donoho et al., 2006; Obozinski et al., 2011) which is more restrictive than RIP (Candes and Tao, 2005; Zhang, 2012). The $\ell_1 + \ell_{1,\infty}$ regularizer is a convex relaxation of an ℓ_0-type one, in which a globally optimal solution can be obtained. However, a convex regularizer is known to be too loose to approximate the ℓ_0-type one and often achieves suboptimal performance (either require restrictive conditions or obtain a suboptimal error bound) (Zou and Li, 2008; Zhang, 2010, 2012; Zhang and Zhang, 2012; Shen et al., 2012; Fan et al., 2012). To remedy the limitation, a non-convex regularizer can be used instead. However, the non-convex formulation is usually difficult to solve and a globally optimal solution can not be obtained in most practical problems. Moreover, the solution of the non-convex formulation heavily depends on the specific optimization algorithms employed. Even with the same optimization algorithm adopted, different initializations usually lead to different solutions. Thus, it is often challenging to analyze the theoretical behavior of a non-convex formulation.

We propose a non-convex formulation, called capped-ℓ_1,ℓ_1 regularized model for multi-task feature learning. The proposed model aims to simultaneously learn the features specific to each task as well as the common features shared among tasks. We propose a Multi-Stage Multi-Task Feature Learning (MSMTFL) algorithm to solve the non-convex optimization problem. We also provide intuitive interpretations of the proposed algorithm from several aspects. In addition, we present a detailed convergence analysis for the proposed algorithm. To address the reproducibility issue of the non-convex formulation, we show that the solution generated by the MSMTFL algorithm is unique (i.e., the solution is reproducible) under a mild condition, which facilitates the theoretical analysis of the MSMTFL algorithm. Although the MSMTFL algorithm may not obtain a globally optimal solution, we show that this solution achieves good performance. Specifically, we present a detailed theoretical analysis on the parameter estimation error bound for the MSMTFL algorithm. Our analysis shows that, under the sparse eigenvalue condition which is *weaker* than the incoherence condition used in Jalali et al. (2010), MSMTFL improves the error bound during the multi-stage iteration, that is, the error bound at the current iteration improves the one at the last iteration. Empirical studies on both synthetic and real-world data sets demonstrate the effectiveness of the MSMTFL algorithm in comparison with the state of the art algorithms.

1.1 Notations and Organization

Scalars and vectors are denoted by lower case letters and bold face lower case letters, respectively. Matrices and sets are denoted by capital letters and calligraphic capital letters, respectively. The ℓ_1 norm, Euclidean norm, ℓ_∞ norm and Frobenius norm are denoted by $\|\cdot\|_1$, $\|\cdot\|$, $\|\cdot\|_\infty$ and $\|\cdot\|_F$, respectively. $|\cdot|$ denotes the absolute value of a scalar or the number of elements in a set, depending on the context. We define the $\ell_{p,q}$ norm of a matrix X as $\|X\|_{p,q} = \left(\sum_i \left((\sum_j |x_{ij}|^q)^{1/q}\right)^p\right)^{1/p}$. We define $\mathbb{N}_n$ as $\{1,\cdots,n\}$ and $N(\mu,\sigma^2)$ as the normal distribution with mean μ and variance σ^2. For a $d\times m$ matrix W and sets $I_i \subseteq \mathbb{N}_d \times \{i\}$, $I \subseteq \mathbb{N}_d \times \mathbb{N}_d$, we let $\mathbf{w}_{I_i}$ be the $d\times 1$ vector with the j-th entry being w_{ji}, if $(j,i) \in I_i$, and 0, otherwise. We also let W_I be a $d\times m$ matrix with the (j,i)-th entry being w_{ji}, if $(j,i) \in I$, and 0, otherwise.

In Section 2, we introduce a non-convex formulation and present the corresponding optimization algorithm. In Section 3, we discuss the convergence and reproducibility issues of the MSMTFL algorithm. In Section 4, we present a detailed theoretical analysis on the MSMTFL algorithm, in terms of the parameter estimation error bound. In Section 5, we provide a sketch of the proof of the presented theoretical results (the detailed proof is provided in the Appendix). In Section 6, we report the experimental results and we conclude the paper in Section 7.

2. The Proposed Formulation and Algorithm

In this section, we first propose a non-convex formulation for multi-task feature learning, based on the capped-ℓ_1,ℓ_1 regularization. Then, we show how to solve the corresponding non-convex optimization problem. Finally, we provide intuitive interpretations and discussions for the proposed algorithm.

2.1 A Non-convex Formulation

Assume we are given m learning tasks associated with training data $\{(X_1,\mathbf{y}_1),\cdots,(X_m,\mathbf{y}_m)\}$, where $X_i \in \mathbb{R}^{n_i\times d}$ is the data matrix of the i-th task with each row as a sample; $\mathbf{y}_i \in \mathbb{R}^{n_i}$ is the response of the i-th task; d is the data dimensionality; n_i is the number of samples for the i-th task. We consider learning a weight matrix $W = [\mathbf{w}_1,\cdots,\mathbf{w}_m] \in \mathbb{R}^{d\times m}$ ($\mathbf{w}_i \in \mathbb{R}^d, i \in \mathbb{N}_m$) consisting of the weight vectors for m linear predictive models: $\mathbf{y}_i \approx \mathbf{f}_i(X_i) = X_i\mathbf{w}_i,\ i \in \mathbb{N}_m$. In this paper, we propose a non-convex multi-task feature learning formulation to learn these m models simultaneously, based on the capped-ℓ_1,ℓ_1 regularization. Specifically, we first impose the ℓ_1 penalty on each row of W, obtaining a column vector. Then, we impose the capped-ℓ_1 penalty (Zhang, 2010, 2012) on that vector. Formally, we formulate our proposed model as follows:

$$\text{capped}-\ell_1,\ell_1: \quad \min_{W\in\mathbb{R}^{d\times m}} \left\{ l(W) + \lambda \sum_{j=1}^{d} \min\left(\|\mathbf{w}^j\|_1, \theta\right) \right\}, \tag{1}$$

where $l(W)$ is an empirical loss function of W; λ (>0) is a parameter balancing the empirical loss and the regularization; θ (>0) is a thresholding parameter; $\mathbf{w}^j$ is the j-th row of the matrix W. In this paper, we focus on the following quadratic loss function:

$$l(W) = \sum_{i=1}^{m} \frac{1}{mn_i} \|X_i\mathbf{w}_i - \mathbf{y}_i\|^2. \tag{2}$$

Intuitively, due to the capped-ℓ_1,ℓ_1 penalty, the optimal solution of Equation (1) denoted as $W^\star$ has many zero rows. For a nonzero row $(\mathbf{w}^\star)^k$, some entries may be zero, due to the ℓ_1-norm imposed on each row of W. Thus, under the formulation in Equation (1), some features can be shared by some tasks but not all the tasks. Therefore, the proposed formulation can leverage the common features shared among tasks.

2.2 Two Relevant Non-convex Formulations

In this subsection, we discuss two relevant non-convex formulations. The first one is the capped-ℓ_1 feature learning formulation:

$$\text{capped}-\ell_1: \quad \min_{W\in\mathbb{R}^{d\times m}} \left\{ l(W) + \lambda \sum_{j=1}^{d}\sum_{i=1}^{m} \min\left(|w_{ji}|, \theta\right) \right\}. \tag{3}$$

Although the optimal solution of formulation (3) has a similar sparse pattern to that of the proposed capped-ℓ_1,ℓ_1 multi-task feature learning (i.e., the optimal solution can have many zero rows and enable some entries of a non-zero row to be zero), the models for different tasks decouple and thus formulation (3) is equivalent to the single task feature learning. Thus, the existing analysis for the single task setting in Zhang (2010, 2012) can be trivially adapted to this setting. The second one is the capped-ℓ_1,ℓ_2 multi-task feature learning formulation:

$$\text{capped}-\ell_1,\ell_2: \quad \min_{W\in\mathbb{R}^{d\times m}} \left\{ l(W) + \lambda \sum_{j=1}^{d} \min\left(\|\mathbf{w}^j\|, \theta\right) \right\}. \tag{4}$$

Due to the use of the capped-ℓ_1,ℓ_2 penalty, the optimal solution $W^\star$ of formulation (4) has many zero rows. However, any non-zero row of $W^\star$ is less likely to contain zero entries because of the Euclidean norm imposed on the rows of W. In other words, each row of $W^\star$ is either a zero vector or a vector composed of all non-zero entries. Thus, in this setting, some relevant features are required to be shared by all tasks. This is obviously different from the motivation of the proposed capped-ℓ_1,ℓ_1 multi-task feature learning, that is, some features are shared by some tasks but not all the tasks.

2.3 Optimization Algorithm

The formulation in Equation (1) is non-convex and is difficult to solve. In this paper, we propose an algorithm called Multi-Stage Multi-Task Feature Learning (MSMTFL) to solve the optimization problem (see details in Algorithm 1).[1] In this algorithm, a key step is how to efficiently solve Equation (5). Observing that the objective function in Equation (5) can be decomposed into the sum of a differential loss function and a non-differential regularization term, we employ FISTA (Beck and Teboulle, 2009) to solve the sub-problem. In the following, we present some intuitive interpretations of the proposed algorithm from several aspects.

1. We can use MSMTFL-type algorithms to solve the non-convex multi-task feature learning problems in Eqs. (3) and (4). Please refer to Appendix C for details.

Algorithm 1: MSMTFL: Multi-Stage Multi-Task Feature Learning

Initialize $\lambda_j^{(0)} = \lambda$;
for $\ell = 1, 2, \cdots$ **do**
  Let $\hat{W}^{(\ell)}$ be a solution of the following problem:

$$\min_{W \in \mathbb{R}^{d \times m}} \left\{ l(W) + \sum_{j=1}^{d} \lambda_j^{(\ell-1)} \|\mathbf{w}^j\|_1 \right\}. \tag{5}$$

  Let $\lambda_j^{(\ell)} = \lambda I(\|(\hat{\mathbf{w}}^{(\ell)})^j\|_1 < \theta)$ $(j = 1, \cdots, d)$, where $(\hat{\mathbf{w}}^{(\ell)})^j$ is the j-th row of $\hat{W}^{(\ell)}$ and $I(\cdot)$ denotes the $\{0,1\}$-valued indicator function.
end

2.3.1 LOCALLY LINEAR APPROXIMATION

First, we define two auxiliary functions:

$$\mathbf{h} : \mathbb{R}^{d \times m} \mapsto \mathbb{R}_+^d,\ \mathbf{h}(W) = \left[\|\mathbf{w}^1\|_1, \cdots, \|\mathbf{w}^d\|_1 \right]^T,$$
$$g : \mathbb{R}_+^d \mapsto \mathbb{R}_+,\ g(\mathbf{u}) = \sum_{j=1}^{d} \min(u_j, \theta).$$

We note that $g(\cdot)$ is a concave function and we say that a vector $\mathbf{s} \in \mathbb{R}^d$ is a sub-gradient of g at $\mathbf{v} \in \mathbb{R}_+^d$, if for all vector $\mathbf{u} \in \mathbb{R}_+^d$, the following inequality holds:

$$g(\mathbf{u}) \leq g(\mathbf{v}) + \langle \mathbf{s}, \mathbf{u} - \mathbf{v} \rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. Using the functions defined above, Equation (1) can be equivalently rewritten as follows:

$$\min_{W \in \mathbb{R}^{d \times m}} \{ l(W) + \lambda g(\mathbf{h}(W)) \}. \tag{6}$$

Based on the definition of the sub-gradient for a concave function given above, we can obtain an upper bound of $g(\mathbf{h}(W))$ using a locally linear approximation at $\mathbf{h}(\hat{W}^{(\ell)})$:

$$g(\mathbf{h}(W)) \leq g(\mathbf{h}(\hat{W}^{(\ell)})) + \left\langle \mathbf{s}^{(\ell)}, \mathbf{h}(W) - \mathbf{h}(\hat{W}^{(\ell)}) \right\rangle,$$

where $\mathbf{s}^{(\ell)}$ is a sub-gradient of $g(\mathbf{u})$ at $\mathbf{u} = \mathbf{h}(\hat{W}^{(\ell)})$. Furthermore, we can obtain an upper bound of the objective function in Equation (6), if the solution $\hat{W}^{(\ell)}$ at the ℓ-th iteration is available:

$$\forall W \in \mathbb{R}^{d \times m} : l(W) + \lambda g(\mathbf{h}(W)) \leq l(W) + \lambda g(\mathbf{h}(\hat{W}^{(\ell)})) + \lambda \left\langle \mathbf{s}^{(\ell)}, \mathbf{h}(W) - \mathbf{h}(\hat{W}^{(\ell)}) \right\rangle. \tag{7}$$

It can be shown that a sub-gradient of $g(\mathbf{u})$ at $\mathbf{u} = \mathbf{h}(\hat{W}^{(\ell)})$ is

$$\mathbf{s}^{(\ell)} = \left[I(\|(\hat{\mathbf{w}}^{(\ell)})^1\|_1 < \theta), \cdots, I(\|(\hat{\mathbf{w}}^{(\ell)})^d\|_1 < \theta) \right]^T, \tag{8}$$

which is used in Step 4 of Algorithm 1. Since both λ and $\mathbf{h}(\hat{W}^{(\ell)})$ are constant with respect to W, we have

$$\begin{aligned}\hat{W}^{(\ell+1)} &= \arg\min_{W} \left\{ l(W) + \lambda g(\mathbf{h}(\hat{W}^{(\ell)})) + \lambda \left\langle \mathbf{s}^{(\ell)}, \mathbf{h}(W) - \mathbf{h}(\hat{W}^{(\ell)}) \right\rangle \right\} \\ &= \arg\min_{W} \left\{ l(W) + \lambda (\mathbf{s}^{(\ell)})^T \mathbf{h}(W) \right\},\end{aligned}$$

which, as shown in Step 3 of Algorithm 1, obtains the next iterative solution by minimizing the upper bound of the objective function in Equation (6). Thus, in the viewpoint of the locally linear approximation, we can understand Algorithm 1 as follows: The original formulation in Equation (6) is non-convex and is difficult to solve; the proposed algorithm minimizes an upper bound in each step, which is convex and can be solved efficiently. It is closely related to the Concave Convex Procedure (CCCP) (Yuille and Rangarajan, 2003). In addition, we can easily verify that the objective function value decreases monotonically as follows:

$$\begin{aligned}l(\hat{W}^{(\ell+1)}) + \lambda g(\mathbf{h}(\hat{W}^{(\ell+1)})) &\leq l(\hat{W}^{(\ell+1)}) + \lambda g(\mathbf{h}(\hat{W}^{(\ell)})) + \lambda \left\langle \mathbf{s}^{(\ell)}, \mathbf{h}(\hat{W}^{(\ell+1)}) - \mathbf{h}(\hat{W}^{(\ell)}) \right\rangle \\ &\leq l(\hat{W}^{(\ell)}) + \lambda g(\mathbf{h}(\hat{W}^{(\ell)})) + \lambda \left\langle \mathbf{s}^{(\ell)}, \mathbf{h}(\hat{W}^{(\ell)}) - \mathbf{h}(\hat{W}^{(\ell)}) \right\rangle \\ &= l(\hat{W}^{(\ell)}) + \lambda g(\mathbf{h}(\hat{W}^{(\ell)})),\end{aligned}$$

where the first inequality is due to Equation (7) and the second inequality follows from the fact that $\hat{W}^{(\ell+1)}$ is a minimizer of the right hand side of Equation (7).

An important issue we should mention is that a monotonic decrease of the objective function value does not guarantee the convergence of the algorithm, even if the objective function is strictly convex and continuously differentiable (see an example in the book (Bertsekas, 1999, Fig 1.2.6)). In Section 3.1, we will formally discuss the convergence issue.

2.3.2 BLOCK COORDINATE DESCENT

Recall that $g(\mathbf{u})$ is a concave function. We can define its conjugate function as (Rockafellar, 1970):

$$g^\star(\mathbf{v}) = \inf_{\mathbf{u}} \{\mathbf{v}^T \mathbf{u} - g(\mathbf{u})\}.$$

Since $g(\mathbf{u})$ is also a closed function (i.e., the epigraph of $g(\mathbf{u})$ is convex), the conjugate function of $g^\star(\mathbf{v})$ is the original function $g(\mathbf{u})$ (Bertsekas, 1999, Chap. 5.4), that is:

$$g(\mathbf{u}) = \inf_{\mathbf{v}} \{\mathbf{u}^T \mathbf{v} - g^\star(\mathbf{v})\}. \tag{9}$$

Substituting Equation (9) with $\mathbf{u} = \mathbf{h}(W)$ into Equation (6), we can reformulate Equation (6) as:

$$\min_{W,\mathbf{v}} \left\{ f(W, \mathbf{v}) = l(W) + \lambda \mathbf{v}^T \mathbf{h}(W) - \lambda g^\star(\mathbf{v}) \right\} \tag{10}$$

A straightforward algorithm for optimizing Equation (10) is the block coordinate descent (Grippo and Sciandrone, 2000; Tseng, 2001) summarized below:

- Fix $W = \hat{W}^{(\ell)}$:

$$\hat{\mathbf{v}}^{(\ell)} = \arg\min_{\mathbf{v}} \left\{ l(\hat{W}^{(\ell)}) + \lambda \mathbf{v}^T \mathbf{h}(\hat{W}^{(\ell)}) - \lambda g^\star(\mathbf{v}) \right\}$$
$$= \arg\min_{\mathbf{v}} \left\{ \mathbf{v}^T \mathbf{h}(\hat{W}^{(\ell)}) - g^\star(\mathbf{v}) \right\}. \tag{11}$$

  Based on Equation (9) and the Danskin's Theorem (Bertsekas, 1999, Proposition B.25), one solution of Equation (11) is given by a sub-gradient of $g(\mathbf{u})$ at $\mathbf{u} = \mathbf{h}(\hat{W}^{(\ell)})$. That is, we can choose $\hat{\mathbf{v}}^{(\ell)} = \mathbf{s}^{(\ell)}$ given in Equation (8). Apparently, Equation (11) is equivalent to Step 4 in Algorithm 1.

- Fix $\mathbf{v} = \hat{\mathbf{v}}^{(\ell)} = \left[I(\|(\hat{\mathbf{w}}^{(\ell)})^1\|_1 < \theta), \cdots, I(\|(\hat{\mathbf{w}}^{(\ell)})^d\|_1 < \theta) \right]^T$:

$$\hat{W}^{(\ell+1)} = \arg\min_{W} \left\{ l(W) + \lambda (\hat{\mathbf{v}}^{(\ell)})^T \mathbf{h}(W) - \lambda g^\star(\hat{\mathbf{v}}^{(\ell)}) \right\}$$
$$= \arg\min_{W} \left\{ l(W) + \lambda (\hat{\mathbf{v}}^{(\ell)})^T \mathbf{h}(W) \right\}, \tag{12}$$

  which corresponds to Step 3 of Algorithm 1.

The block coordinate descent procedure is intuitive, however, it is non-trivial to analyze its convergence behavior. We will present the convergence analysis in Section 3.1.

2.3.3 DISCUSSIONS

If we terminate the algorithm with $\ell = 1$, the MSMTFL algorithm is equivalent to the ℓ_1 regularized multi-task feature learning algorithm (Lasso). Thus, the solution obtained by MSMTFL can be considered as a multi-stage refinement of that of Lasso. Basically, the MSMTFL algorithm solves a sequence of weighted Lasso problems, where the weights λ_j's are set as the product of the parameter λ in Equation (1) and a $\{0,1\}$-valued indicator function. Specifically, a penalty is imposed in the current stage if the ℓ_1-norm of some row of W in the last stage is smaller than the threshold θ; otherwise, no penalty is imposed. In other words, MSMTFL in the current stage tends to shrink the small rows of W and keep the large rows of W in the last stage. However, Lasso (corresponds to $\ell = 1$) penalizes all rows of W in the same way. It may incorrectly keep the irrelevant rows (which should have been zero rows) or shrink the relevant rows (which should have been large rows) to be zero vectors. MSMTFL overcomes this limitation by adaptively penalizing the rows of W according to the solution generated in the last stage. One important question is whether the MSMTFL algorithm can improve the performance during the multi-stage iteration. In Section 4, we will theoretically show that the MSMTFL algorithm indeed achieves the stagewise improvement in terms of the parameter estimation error bound. That is, the error bound in the current stage improves the one in the last stage. Empirical studies in Section 6 also validate the presented theoretical analysis.

3. Convergence and Reproducibility Analysis

In this section, we first present the convergence analysis. Then, we discuss the reproducibility issue for the MSMTFL algorithm.

3.1 Convergence Analysis

The main convergence result is summarized in the following theorem, which is based on the block coordinate descent interpretation.

Theorem 1 *Let $(W^\star, \mathbf{v}^\star)$ be a limit point of the sequence $\{\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}\}$ generated by the block coordinate descent algorithm. Then $W^\star$ is a critical point of Equation (1).*

Proof Based on Equation (11) and Equation (12), we have

$$\begin{aligned} f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}) &\leq f(\hat{W}^{(\ell)}, \mathbf{v}),\ \forall \mathbf{v} \in \mathbb{R}^d, \\ f(\hat{W}^{(\ell+1)}, \hat{\mathbf{v}}^{(\ell)}) &\leq f(W, \hat{\mathbf{v}}^{(\ell)}),\ \forall W \in \mathbb{R}^{d\times m}. \end{aligned} \tag{13}$$

It follows that

$$f(\hat{W}^{(\ell+1)}, \hat{\mathbf{v}}^{(\ell+1)}) \leq f(\hat{W}^{(\ell+1)}, \hat{\mathbf{v}}^{(\ell)}) \leq f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}),$$

which indicates that the sequence $\{f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)})\}$ is monotonically decreasing. Since $(W^\star, \mathbf{v}^\star)$ is a limit point of $\{\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}\}$, there exists a subsequence $\mathcal{K}$ such that

$$\lim_{\ell \in \mathcal{K} \to \infty} (\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}) = (W^\star, \mathbf{v}^\star).$$

We observe that

$$\begin{aligned} f(W, \mathbf{v}) &= l(W) + \lambda \mathbf{v}^T \mathbf{h}(W) - \lambda g^\star(\mathbf{v}) \\ &\geq l(W) + \lambda g(\mathbf{h}(W)) \geq 0, \end{aligned}$$

where the first inequality above is due to Equation (9). Thus, $\{f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)})\}_{\ell \in \mathcal{K}}$ is bounded below. Together with the fact that $\{f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)})\}$ is decreasing, $\lim_{\ell\to\infty} f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}) > -\infty$ exists. Since $f(W, \mathbf{v})$ is continuous, we have

$$\lim_{\ell\to\infty} f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}) = \lim_{\ell \in \mathcal{K} \to \infty} f(\hat{W}^{(\ell)}, \hat{\mathbf{v}}^{(\ell)}) = f(W^\star, \mathbf{v}^\star).$$

Taking limits on both sides of Equation (13) with $\ell \in \mathcal{K} \to \infty$, we have

$$f(W^\star, \mathbf{v}^\star) \leq f(W, \mathbf{v}^\star),\ \forall W \in \mathbb{R}^{d\times m},$$

which implies

$$\begin{aligned} W^\star &\in \arg\min_W f(W, \mathbf{v}^\star) \\ &= \arg\min_W \left\{ l(W) + \lambda (\mathbf{v}^\star)^T \mathbf{h}(W) - \lambda g^\star(\mathbf{v}^\star) \right\} \\ &= \arg\min_W \left\{ l(W) + \lambda (\mathbf{v}^\star)^T \mathbf{h}(W) \right\}. \end{aligned} \tag{14}$$

Therefore, the zero matrix O must be a sub-gradient of the objective function in Equation (14) at $W = W^\star$:

$$O \in \partial l(W^\star) + \lambda \partial \left((\mathbf{v}^\star)^T \mathbf{h}(W^\star) \right) = \partial l(W^\star) + \lambda \sum_{j=1}^{d} v_j^\star \partial \left(\|(\mathbf{w}^\star)^j\|_1 \right), \tag{15}$$

where $\partial l(W^\star)$ denotes the sub-differential (which is a set composed of all sub-gradients) of $l(W)$ at $W = W^\star$. We observe that

$$\hat{\mathbf{v}}^{(\ell)} \in \partial g(\mathbf{u})|_{\mathbf{u}=\mathbf{h}(\hat{W}^{(\ell)})},$$

which implies that $\forall \mathbf{x} \in \mathbb{R}^d_+$:

$$g(\mathbf{x}) \leq g(\mathbf{h}(\hat{W}^{(\ell)})) + \left\langle \hat{\mathbf{v}}^{(\ell)}, \mathbf{x} - \mathbf{h}(\hat{W}^{(\ell)}) \right\rangle.$$

Taking limits on both sides of the above inequality with $\ell \in \mathcal{K} \to \infty$, we have:

$$g(\mathbf{x}) \leq g(\mathbf{h}(W^\star)) + \langle \mathbf{v}^\star, \mathbf{x} - \mathbf{h}(W^\star) \rangle,$$

which implies that $\mathbf{v}^\star$ is a sub-gradient of $g(\mathbf{u})$ at $\mathbf{u} = \mathbf{h}(W^\star)$, that is:

$$\mathbf{v}^\star \in \partial g(\mathbf{u})|_{\mathbf{u}=\mathbf{h}(W^\star)}. \tag{16}$$

Note that the objective function of Equation (1) can be written as a difference of two convex functions:

$$l(W) + \lambda \sum_{j=1}^{d} \min\left(\|\mathbf{w}^j\|_1, \theta\right) = \left(l(W) + \lambda \|W\|_{1,1}\right) - \lambda \sum_{j=1}^{d} \max\left(\|\mathbf{w}^j\|_1 - \theta, 0\right).$$

Based on Wright et al. (2009); Toland (1979), we know that $W^\star$ is a critical point of Equation (1) if the following holds:

$$O \in \left(\nabla l(W^\star) + \lambda \partial \|W^\star\|_{1,1}\right) - \lambda \sum_{j=1}^{d} \partial \max\left(\|(\mathbf{w}^\star)^j\|_1 - \theta, 0\right). \tag{17}$$

Substituting Equation (16) into Equation (15), we can obtain Equation (17). Therefore, $W^\star$ is a critical point of Equation (1). This completes the proof of Theorem 1. ■

Due to the equivalence between Algorithm 1 and the block coordinate descent algorithm above, Theorem 1 indicates that any limit point of the sequence $\{\hat{W}^{(\ell)}\}$ generated by Algorithm 1 is a critical point of Equation (1). The remaining issue is to analyze the performance of the critical point. In the sequel, we will conduct analysis in two aspects: reproducibility and the parameter estimation performance.

3.2 Reproducibility of The Algorithm

In general, it is difficult to analyze the performance of a non-convex formulation, as different solutions can be obtained due to different initializations. One natural question is whether the solution generated by Algorithm 1 (based on the initialization of $\lambda_j^{(0)} = \lambda$ in Step 1) is reproducible. In other words, is the solution of Algorithm 1 unique? If we can guarantee that, for any $\ell \geq 1$, the solution $\hat{W}^{(\ell)}$ of Equation (5) is unique, then the solution generated by Algorithm 1 is unique. That is, the solution is reproducible. The main result is summarized in the following theorem:

Theorem 2 *If $X_i \in \mathbb{R}^{n_i \times d}$ $(i \in \mathbb{N}_m)$ has entries drawn from a continuous probability distribution on $R^{n_i d}$, then, for any $\ell \geq 1$, the optimization problem in Equation (5) has a unique solution with probability one.*

Proof Equation (5) can be decomposed into m independent smaller minimization problems:

$$\hat{\mathbf{w}}_i^{(\ell)} = \arg\min_{\mathbf{w}_i \in \mathbb{R}^d} \frac{1}{mn_i} \|X_i \mathbf{w}_i - \mathbf{y}_i\|^2 + \sum_{j=1}^{d} \lambda_j^{(\ell-1)} |w_{ji}|.$$

Next, we only need to prove that the solution of the above optimization problem is unique. To simplify the notations, we unclutter the above equation (by ignoring some superscripts and subscripts) as follows:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{mn} \|X\mathbf{w} - \mathbf{y}\|^2 + \sum_{j=1}^{d} \lambda_j |w_j|, \tag{18}$$

The first order optimal condition is $\forall j \in \mathbb{N}_d$:

$$\frac{2}{mn} \mathbf{x}_j^T (\mathbf{y} - X\hat{\mathbf{w}}) = \lambda_j \text{sign}(\hat{w}_j),$$

where $\text{sign}(\hat{w}_j) = 1$, if $\hat{w}_j > 0$; $\text{sign}(\hat{w}_j) = -1$, if $\hat{w}_j < 0$; and $\text{sign}(\hat{w}_j) \in [-1, 1]$, otherwise. We define

$$\mathcal{E} = \left\{ j \in \mathbb{N}_d : \frac{2}{mn} |\mathbf{x}_j^T (\mathbf{y} - X\hat{\mathbf{w}})| = \lambda_j \right\},$$
$$\mathbf{s} = \text{sign}\left(\frac{2}{mn} X_{\mathcal{E}}^T (\mathbf{y} - X\hat{\mathbf{w}}) \right),$$

where $X_{\mathcal{E}}$ denotes the matrix composed of the columns of X indexed by $\mathcal{E}$. Then, the optimal solution $\hat{\mathbf{w}}$ of Equation (18) satisfies

$$\hat{\mathbf{w}}_{\mathbb{N}_d \setminus \mathcal{E}} = \mathbf{0},$$
$$\hat{\mathbf{w}}_{\mathcal{E}} = \arg\min_{\mathbf{w}_{\mathcal{E}} \in \mathbb{R}^{|\mathcal{E}|}} \frac{1}{mn} \|X_{\mathcal{E}} \mathbf{w}_{\mathcal{E}} - \mathbf{y}\|^2 + \sum_{j \in \mathcal{E}} \lambda_j |w_j|, \tag{19}$$

where $\mathbf{w}_{\mathcal{E}}$ denotes the vector composed of entries of $\mathbf{w}$ indexed by $\mathcal{E}$. Since $X \in \mathbb{R}^{n_i \times d}$ is drawn from the continuous probability distribution, X has columns in general positions with probability one and hence $\text{rank}(X_{\mathcal{E}}) = |\mathcal{E}|$ (or equivalently $\text{Null}(X_{\mathcal{E}}) = \{\mathbf{0}\}$), due to Lemma 3, Lemma 4 and their discussions in Tibshirani (2013). Therefore, the objective function in Equation (19) is strictly convex. Noticing that $X\hat{\mathbf{w}}$ is unique (Tibshirani, 2013), thus $\mathcal{E}$ is unique. This implies that $\hat{\mathbf{w}}_{\mathcal{E}}$ is unique. Thus, the optimal solution $\hat{\mathbf{w}}$ of Equation (18) is also unique and so is the optimization problem in Equation (5) for any $\ell \geq 1$. This completes the proof of Theorem 2. ■

Theorem 2 is important in the sense that it makes the theoretical analysis for the parameter estimation performance of Algorithm 1 possible. Although the solution may not be globally optimal, we show in the next section that the solution has good performance in terms of the parameter estimation error bound.

Remark 3 *Zhang (2010, 2012) study the capped-ℓ_1 regularized formulation for the single task setting and propose the multi-stage algorithm for such formulation. However, Zhang (2010, 2012) neither provide detailed convergence analysis nor discuss the reproducibility issues. The presented analysis is applicable to the multi-stage algorithm proposed in Zhang (2010, 2012), as it is a special case of the proposed algorithm with $m = 1$. To our best knowledge, this is the first work that discusses the reproducibility issue for multi-stage optimization algorithms.*

4. Parameter Estimation Error Bound

In this section, we theoretically analyze the parameter estimation performance of the solution obtained by the MSMTFL algorithm. To simplify the notations in the theoretical analysis, we assume that the number of samples for all the tasks are the same. However, our theoretical analysis can be easily extended to the case where the tasks have different sample sizes.

We first present a sub-Gaussian noise assumption which is very common in the analysis of sparse learning literature (Zhang and Zhang, 2012; Zhang, 2008, 2009, 2010, 2012).

Assumption 1 *Let $\bar{W} = [\bar{\mathbf{w}}_1, \cdots, \bar{\mathbf{w}}_m] \in \mathbb{R}^{d\times m}$ be the underlying sparse weight matrix and $\mathbf{y}_i = X_i\bar{\mathbf{w}}_i + \boldsymbol{\delta}_i$, $\mathbb{E}\mathbf{y}_i = X_i\bar{\mathbf{w}}_i$, where $\boldsymbol{\delta}_i \in \mathbb{R}^n$ is a random vector with all entries δ_{ji} $(j \in \mathbb{N}_n, i \in \mathbb{N}_m)$ being independent sub-Gaussians: there exists $\sigma > 0$ such that $\forall j \in \mathbb{N}_n, i \in \mathbb{N}_m, t \in \mathbb{R}$:*

$$\mathbb{E}_{\delta_{ji}} \exp(t\delta_{ji}) \le \exp\left(\frac{\sigma^2 t^2}{2}\right).$$

Remark 4 *We call the random variable satisfying the condition in Assumption 1 sub-Gaussian, since its moment generating function is bounded by that of a zero mean Gaussian random variable. That is, if a normal random variable $x \sim N(0,\sigma^2)$, then we have:*

$$\begin{aligned}\mathbb{E}\exp(tx) &= \int_{-\infty}^{\infty} \exp(tx)\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{x^2}{2\sigma^2}\right)dx \\ &= \exp(\sigma^2t^2/2)\int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\sigma^2t)^2}{2\sigma^2}\right)dx \\ &= \exp(\sigma^2t^2/2).\end{aligned}$$

Remark 5 *Based on the Hoeffding's Lemma, for any random variable $x \in [a,b]$ and $\mathbb{E}x = 0$, we have $\mathbb{E}(\exp(tx)) \le \exp\left(\frac{t^2(b-a)^2}{8}\right)$. Therefore, both zero mean Gaussian and zero mean bounded random variables are sub-Gaussians. Thus, the sub-Gaussian noise assumption is more general than the Gaussian noise assumption which is commonly used in the multi-task learning literature (Jalali et al., 2010; Lounici et al., 2009).*

We next introduce the following sparse eigenvalue concept which is also common in the analysis of sparse learning literature (Zhang and Huang, 2008; Zhang and Zhang, 2012; Zhang, 2009, 2010, 2012).

Definition 6 *Given $1 \le k \le d$, we define*

$$\rho_i^+(k) = \sup_{\mathbf{w}}\left\{\frac{\|X_i\mathbf{w}\|^2}{n\|\mathbf{w}\|^2} : \|\mathbf{w}\|_0 \le k\right\},\ \rho_{max}^+(k) = \max_{i\in\mathbb{N}_m}\rho_i^+(k),$$
$$\rho_i^-(k) = \inf_{\mathbf{w}}\left\{\frac{\|X_i\mathbf{w}\|^2}{n\|\mathbf{w}\|^2} : \|\mathbf{w}\|_0 \le k\right\},\ \rho_{min}^-(k) = \min_{i\in\mathbb{N}_m}\rho_i^-(k).$$

Remark 7 *$\rho_i^+(k)$ ($\rho_i^-(k)$) is in fact the maximum (minimum) eigenvalue of $(X_i)_{\mathcal{S}}^T(X_i)_{\mathcal{S}}/n$, where $\mathcal{S}$ is a set satisfying $|\mathcal{S}| \leq k$ and $(X_i)_{\mathcal{S}}$ is a submatrix composed of the columns of X_i indexed by $\mathcal{S}$. In the MTL setting, we need to exploit the relations of $\rho_i^+(k)$ ($\rho_i^-(k)$) among multiple tasks.*

We present our parameter estimation error bound on MSMTFL in the following theorem:

Theorem 8 *Let Assumption 1 hold. Define $\bar{\mathcal{F}}_i = \{(j,i) : \bar{w}_{ji} \neq 0\}$ and $\bar{\mathcal{F}} = \cup_{i \in \mathbb{N}_m} \bar{\mathcal{F}}_i$. Denote $\bar{r}$ as the number of nonzero rows of $\bar{W}$. We assume that*

$$\forall (j,i) \in \bar{\mathcal{F}}, \|\bar{\mathbf{w}}^j\|_1 \geq 2\theta \tag{20}$$

$$\text{and } \frac{\rho_i^+(s)}{\rho_i^-(2\bar{r}+2s)} \leq 1 + \frac{s}{2\bar{r}}, \tag{21}$$

where s is some integer satisfying $s \geq \bar{r}$. If we choose λ and θ such that for some $s \geq \bar{r}$:

$$\lambda \geq 12\sigma\sqrt{\frac{2\rho_{max}^+(1)\ln(2dm/\eta)}{n}}, \tag{22}$$

$$\theta \geq \frac{11m\lambda}{\rho_{min}^-(2\bar{r}+s)}, \tag{23}$$

then the following parameter estimation error bound holds with probability larger than $1-\eta$:

$$\|\hat{W}^{(\ell)} - \bar{W}\|_{2,1} \leq 0.8^{\ell/2}\frac{9.1m\lambda\sqrt{\bar{r}}}{\rho_{min}^-(2\bar{r}+s)} + \frac{39.5m\sigma\sqrt{\rho_{max}^+(\bar{r})(7.4\bar{r}+2.7\ln(2/\eta))/n}}{\rho_{min}^-(2\bar{r}+s)}, \tag{24}$$

where $\hat{W}^{(\ell)}$ is a solution of Equation (5).

Remark 9 *Equation (20) assumes that the ℓ_1-norm of each nonzero row of $\bar{W}$ is away from zero. This requires the true nonzero coefficients should be large enough, in order to distinguish them from the noise. Equation (21) is called the sparse eigenvalue condition (Zhang, 2012), which requires the eigenvalue ratio $\rho_i^+(s)/\rho_i^-(s)$ to grow sub-linearly with respect to s. Such a condition is very common in the analysis of sparse regularization (Zhang and Huang, 2008; Zhang, 2009) and it is slightly weaker than the RIP condition (Candes and Tao, 2005; Huang and Zhang, 2010; Zhang, 2012).*

Remark 10 *When $\ell = 1$ (corresponds to Lasso), the first term of the right-hand side of Equation (24) dominates the error bound in the order of*

$$\|\hat{W}^{Lasso} - \bar{W}\|_{2,1} = O\left(m\sqrt{\bar{r}\ln(dm/\eta)/n}\right), \tag{25}$$

since λ satisfies the condition in Equation (22). Note that the first term of the right-hand side of Equation (24) shrinks exponentially as ℓ increases. When ℓ is sufficiently large in the order of $O(\ln(m\sqrt{\bar{r}/n}) + \ln\ln(dm))$, this term tends to zero and we obtain the following parameter estimation error bound:

$$\|\hat{W}^{(\ell)} - \bar{W}\|_{2,1} = O\left(m\sqrt{\bar{r}/n + \ln(1/\eta)/n}\right). \tag{26}$$

Jalali et al. (2010) gave an $\ell_{\infty,\infty}$-norm error bound $\|\hat{W}^{Dirty} - \bar{W}\|_{\infty,\infty} = O\left(\sqrt{\ln(dm/\eta)/n}\right)$ as well as a sign consistency result between $\hat{W}$ and $\bar{W}$. A direct comparison between these two bounds is difficult due to the use of different norms. On the other hand, the worst-case estimate of the $\ell_{2,1}$-norm error bound of the algorithm in Jalali et al. (2010) is in the same order with Equation (25), that is: $\|\hat{W}^{Dirty} - \bar{W}\|_{2,1} = O\left(m\sqrt{\bar{r}\ln(dm/\eta)/n}\right)$. When dm is large and the ground truth has a large number of sparse rows (i.e., $\bar{r}$ is a small constant), the bound in Equation (26) is significantly better than the ones for the Lasso and Dirty model.

Remark 11 *Jalali et al. (2010) presented an $\ell_{\infty,\infty}$-norm parameter estimation error bound and hence a sign consistency result can be obtained. The results are derived under the incoherence condition which is more restrictive than the RIP condition and hence more restrictive than the sparse eigenvalue condition in Equation (21). From the viewpoint of the parameter estimation error, our proposed algorithm can achieve a better bound under weaker conditions. Please refer to (Van De Geer and Bühlmann, 2009; Zhang, 2009, 2012) for more details about the incoherence condition, the RIP condition, the sparse eigenvalue condition and their relationships.*

Remark 12 *The capped-ℓ_1 regularized formulation in Zhang (2010) is a special case of our formulation when $m = 1$. However, extending the analysis from the single task to the multi-task setting is nontrivial. Different from previous work on multi-stage sparse learning which focuses on a single task (Zhang, 2010, 2012), we study a more general multi-stage framework in the multi-task setting. We need to exploit the relationship among tasks, by using the relations of sparse eigenvalues $\rho_i^+(k)$ $(\rho_i^-(k))$ and treating the ℓ_1-norm on each row of the weight matrix as a whole for consideration. Moreover, we simultaneously exploit the relations of each column and each row of the matrix.*

In addition, we want to emphasize that the support recovery analysis in Zhang (2012) can not be easily adapted to the proposed capped-ℓ_1,ℓ_1 multi-task feature learning setting. The key difficulty is that, in order to achieve a similar support recovery result for the formulation in Equation (1), we need to assume that each row of the underlying sparse weight matrix $\bar{W}$ is either a zero vector or a vector composed of all nonzero entries. However, this is not the case in the proposed multi-task formulation. Although this assumption holds for the capped-ℓ_1,ℓ_2 multi-task feature learning problem in Equation (4), each subproblem involved for solving Equation (4) is a reweighed ℓ_2 regularized problem and its first-order optimality condition is quite different from that of the reweighed ℓ_1 regularized problem. Thus, it is also challenging to extend the analysis in Zhang (2012) to the capped-ℓ_1,ℓ_2 multi-task feature learning setting.

5. Proof Sketch of Theorem 8

In this section, we present a proof sketch of Theorem 8. We first provide several important lemmas (detailed proofs are available in the Appendix A) and then complete the proof of Theorem 8 based on these lemmas.

Lemma 13 *Let $\bar{\Upsilon} = [\bar{\epsilon}_1, \cdots, \bar{\epsilon}_m]$ with $\bar{\epsilon}_i = [\bar{\varepsilon}_{1i}, \cdots, \bar{\varepsilon}_{di}]^T = \frac{1}{n} X_i^T (X_i \bar{\mathbf{w}}_i - \mathbf{y}_i)$ $(i \in \mathbb{N}_m)$. Define $\bar{\mathcal{H}} \supseteq \bar{\mathcal{F}}$ such that $(j,i) \in \bar{\mathcal{H}}$ $(\forall i \in \mathbb{N}_m)$, provided there exists $(j,g) \in \bar{\mathcal{F}}$ ($\bar{\mathcal{H}}$ is a set consisting of the indices of all entries in the nonzero rows of $\bar{W}$). Under the conditions of Assumption 1 and the notations of*

Theorem 8, the followings hold with probability larger than $1-\eta$*:*

$$\|\bar{\Upsilon}\|_{\infty,\infty} \leq \sigma\sqrt{\frac{2\rho^{+}_{max}(1)\ln(2dm/\eta)}{n}}, \tag{27}$$

$$\|\bar{\Upsilon}_{\bar{\mathcal{H}}}\|_F^2 \leq m\sigma^2\rho^{+}_{max}(\bar{r})(7.4\bar{r}+2.7\ln(2/\eta))/n. \tag{28}$$

Lemma 13 gives bounds on the residual correlation ($\bar{\Upsilon}$) with respect to $\bar{W}$. We note that Equation (27) and Equation (28) are closely related to the assumption on λ in Equation (22) and the second term of the right-hand side of Equation (24) (error bound), respectively. This lemma provides a fundamental basis for the proof of Theorem 8.

Lemma 14 *Use the notations of Lemma 13 and consider* $\mathcal{G}_i \subseteq \mathbb{N}_d \times \{i\}$ *such that* $\bar{\mathcal{F}}_i \cap \mathcal{G}_i = \emptyset$ $(i \in \mathbb{N}_m)$*. Let* $\hat{W} = \hat{W}^{(\ell)}$ *be a solution of Equation (5) and* $\Delta\hat{W} = \hat{W} - \bar{W}$*. Denote* $\hat{\boldsymbol{\lambda}}_i = \hat{\boldsymbol{\lambda}}_i^{(\ell-1)} = [\lambda_1^{(\ell-1)}, \cdots, \lambda_d^{(\ell-1)}]^T$*. Let* $\hat{\lambda}_{\mathcal{G}_i} = \min_{(j,i)\in\mathcal{G}_i}\hat{\lambda}_{ji}$*,* $\hat{\lambda}_{\mathcal{G}} = \min_{i\in\mathbb{N}_m}\hat{\lambda}_{\mathcal{G}_i}$ *and* $\hat{\lambda}_{0i} = \max_j \hat{\lambda}_{ji}$*,* $\hat{\lambda}_0 = \max_i \hat{\lambda}_{0i}$*. If* $2\|\bar{\epsilon}_i\|_\infty < \hat{\lambda}_{\mathcal{G}_i}$*, then the following inequality holds at any stage* $\ell \geq 1$*:*

$$\sum_{i=1}^{m}\sum_{(j,i)\in\mathcal{G}_i}|\hat{w}_{ji}^{(\ell)}| \leq \frac{2\|\bar{\Upsilon}\|_{\infty,\infty}+\hat{\lambda}_0}{\hat{\lambda}_{\mathcal{G}}-2\|\bar{\Upsilon}\|_{\infty,\infty}}\sum_{i=1}^{m}\sum_{(j,i)\in\mathcal{G}_i^c}|\Delta\hat{w}_{ji}^{(\ell)}|.$$

Denote $\mathcal{G} = \cup_{i\in\mathbb{N}_m}\mathcal{G}_i$, $\bar{\mathcal{F}} = \cup_{i\in\mathbb{N}_m}\bar{\mathcal{F}}_i$ and notice that $\bar{\mathcal{F}}\cap\mathcal{G} = \emptyset \Rightarrow \Delta\hat{W}_{\mathcal{G}}^{(\ell)} = \hat{W}_{\mathcal{G}}^{(\ell)}$. Lemma 14 says that $\|\Delta\hat{W}_{\mathcal{G}}^{(\ell)}\|_{1,1} = \|\hat{W}_{\mathcal{G}}^{(\ell)}\|_{1,1}$ is upper bounded in terms of $\|\Delta\hat{W}_{\mathcal{G}^c}^{(\ell)}\|_{1,1}$, which indicates that the error of the estimated coefficients locating outside of $\bar{\mathcal{F}}$ should be small enough. This provides an intuitive explanation why the parameter estimation error of our algorithm can be small.

Lemma 15 *Using the notations of Lemma 14, we denote* $\mathcal{G} = \mathcal{G}_{(\ell)} = \bar{\mathcal{H}}^c \cap \{(j,i) : \hat{\lambda}_{ji}^{(\ell-1)} = \lambda\} = \cup_{i\in\mathbb{N}_m}\mathcal{G}_i$ *with* $\bar{\mathcal{H}}$ *being defined as in Lemma 13 and* $\mathcal{G}_i \subseteq \mathbb{N}_d \times \{i\}$*. Let* $\mathcal{J}_i$ *be the indices of the largest s coefficients (in absolute value) of* $\hat{\mathbf{w}}_{\mathcal{G}_i}$*,* $I_i = \mathcal{G}_i^c \cup \mathcal{J}_i$*,* $I = \cup_{i\in\mathbb{N}_m} I_i$ *and* $\bar{\mathcal{F}} = \cup_{i\in\mathbb{N}_m}\bar{\mathcal{F}}_i$*. Then, the following inequalities hold at any stage* $\ell \geq 1$*:*

$$\|\Delta\hat{W}^{(\ell)}\|_{2,1} \leq \frac{\left(1+1.5\sqrt{\frac{2\bar{r}}{s}}\right)\sqrt{8m\left(4\|\bar{\Upsilon}_{\mathcal{G}_{(\ell)}^c}\|_F^2 + \sum_{(j,i)\in\bar{\mathcal{F}}}(\hat{\lambda}_{ji}^{(\ell-1)})^2\right)}}{\rho^{-}_{min}(2\bar{r}+s)}, \tag{29}$$

$$\|\Delta\hat{W}^{(\ell)}\|_{2,1} \leq \frac{9.1m\lambda\sqrt{\bar{r}}}{\rho^{-}_{min}(2\bar{r}+s)}. \tag{30}$$

Lemma 15 is established based on Lemma 14, by considering the relationship between Equation (22) and Equation (27), and the specific definition of $\mathcal{G} = \mathcal{G}_{(\ell)}$. Equation (29) provides a parameter estimation error bound in terms of $\ell_{2,1}$-norm by $\|\bar{\Upsilon}_{\mathcal{G}_{(\ell)}^c}\|_F^2$ and the regularization parameters $\hat{\lambda}_{ji}^{(\ell-1)}$ (see the definition of $\hat{\lambda}_{ji}$ ($\hat{\lambda}_{ji}^{(\ell-1)}$) in Lemma 14). This is the result directly used in the proof of Theorem 8. Equation (30) states that the error bound is upper bounded in terms of λ, the right-hand side of which constitutes the shrinkage part of the error bound in Equation (24).

Lemma 16 *Let* $\hat{\lambda}_{ji} = \lambda I\left(\|\hat{\mathbf{w}}^j\|_1 < \theta, j \in \mathbb{N}_d\right), \forall i \in \mathbb{N}_m$ *with some* $\hat{W} \in \mathbb{R}^{d\times m}$*.* $\bar{\mathcal{H}} \supseteq \bar{\mathcal{F}}$ *is defined in Lemma 13. Then under the condition of Equation (20), we have:*

$$\sum_{(j,i)\in\bar{\mathcal{F}}}\hat{\lambda}_{ji}^2 \leq \sum_{(j,i)\in\bar{\mathcal{H}}}\hat{\lambda}_{ji}^2 \leq m\lambda^2\|\bar{W}_{\bar{\mathcal{H}}} - \hat{W}_{\bar{\mathcal{H}}}\|_{2,1}^2/\theta^2.$$

Lemma 16 establishes an upper bound of $\sum_{(j,i)\in\bar{\mathcal{F}}}\hat{\lambda}_{ji}^2$ by $\|\bar{W}_{\bar{\mathcal{H}}}-\hat{W}_{\bar{\mathcal{H}}}\|_{2,1}^2$, which is critical for building the recursive relationship between $\|\hat{W}^{(\ell)}-\bar{W}\|_{2,1}$ and $\|\hat{W}^{(\ell-1)}-\bar{W}\|_{2,1}$ in the proof of Theorem 8. This recursive relation is crucial for the shrinkage part of the error bound in Equation (24).

5.1 Proof of Theorem 8

We now complete the proof of Theorem 8 based on the lemmas above.

Proof For notational simplicity, we denote the right-hand side of Equation (28) as:

$$u = m\sigma^2\rho_{max}^+(\bar{r})(7.4\bar{r}+2.7\ln(2/\eta))/n. \tag{31}$$

Based on $\bar{\mathcal{H}}\subseteq\mathcal{G}_{(\ell)}^c$, Lemma 13 and Equation (22), the followings hold with probability larger than $1-\eta$:

$$\begin{aligned}
\|\bar{\Upsilon}_{\mathcal{G}_{(\ell)}^c}\|_F^2 &= \|\bar{\Upsilon}_{\bar{\mathcal{H}}}\|_F^2 + \|\bar{\Upsilon}_{\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}}\|_F^2 \\
&\le u + |\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}|\|\bar{\Upsilon}\|_{\infty,\infty}^2 \\
&\le u + \lambda^2|\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}|/144 \\
&\le u + (1/144)m\lambda^2\theta^{-2}\|\hat{W}_{\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}}^{(\ell-1)} - \bar{W}_{\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}}\|_{2,1}^2,
\end{aligned} \tag{32}$$

where the last inequality follows from

$$\begin{aligned}
&\forall(j,i)\in\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}, \|(\hat{\mathbf{w}}^{(\ell-1)})^j\|_1^2/\theta^2 = \|(\hat{\mathbf{w}}^{(\ell-1)})^j-\bar{\mathbf{w}}^j\|_1^2/\theta^2 \ge 1 \\
\Rightarrow\ &|\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}| \le m\theta^{-2}\|\hat{W}_{\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}}^{(\ell-1)} - \bar{W}_{\mathcal{G}_{(\ell)}^c\setminus\bar{\mathcal{H}}}\|_{2,1}^2.
\end{aligned}$$

According to Equation (29), we have:

$$\begin{aligned}
&\|\hat{W}^{(\ell)}-\bar{W}\|_{2,1}^2 = \|\Delta\hat{W}^{(\ell)}\|_{2,1}^2 \\
&\le \frac{8m\left(1+1.5\sqrt{\frac{2\bar{r}}{s}}\right)^2\left(4\|\bar{\Upsilon}_{\mathcal{G}_{(\ell)}^c}\|_F^2+\sum_{(j,i)\in\bar{\mathcal{F}}}(\hat{\lambda}_{ji}^{(\ell-1)})^2\right)}{(\rho_{min}^-(2\bar{r}+s))^2} \\
&\le \frac{78m\left(4u+(37/36)m\lambda^2\theta^{-2}\left\|\hat{W}^{(\ell-1)}-\bar{W}\right\|_{2,1}^2\right)}{(\rho_{min}^-(2\bar{r}+s))^2} \\
&\le \frac{312mu}{(\rho_{min}^-(2\bar{r}+s))^2} + 0.8\left\|\hat{W}^{(\ell-1)}-\bar{W}\right\|_{2,1}^2 \\
&\le \cdots \le 0.8^\ell\left\|\hat{W}^{(0)}-\bar{W}\right\|_{2,1}^2 + \frac{312mu}{(\rho_{min}^-(2\bar{r}+s))^2}\frac{1-0.8^\ell}{1-0.8} \\
&\le 0.8^\ell\frac{9.1^2m^2\lambda^2\bar{r}}{(\rho_{min}^-(2\bar{r}+s))^2} + \frac{1560mu}{(\rho_{min}^-(2\bar{r}+s))^2}.
\end{aligned}$$

In the above derivation, the first inequality is due to Equation (29); the second inequality is due to the assumption $s\ge\bar{r}$ in Theorem 8, Equation (32) and Lemma 16; the third inequality is due

to Equation (23); the last inequality follows from Equation (30) and $1-0.8^{\ell} \leq 1$ $(\ell \geq 1)$. Thus, following the inequality $\sqrt{a+b} \leq \sqrt{a}+\sqrt{b}$ $(\forall a,b \geq 0)$, we obtain:

$$\|\hat{W}^{(\ell)} - \bar{W}\|_{2,1} \leq 0.8^{\ell/2} \frac{9.1m\lambda\sqrt{\bar{r}}}{\rho^{-}_{min}(2\bar{r}+s)} + \frac{39.5\sqrt{mu}}{\rho^{-}_{min}(2\bar{r}+s)}.$$

Substituting Equation (31) into the above inequality, we verify Theorem 8. ■

Remark 17 *The assumption $s \geq \bar{r}$ used in the above proof indicates that at each stage, the zero entries of $\hat{W}^{(\ell)}$ should be greater than $m\bar{r}$ (see definition of s in Lemma 15). This requires the solution obtained by Algorithm 1 at each stage is sparse, which is consistent with the sparsity of $\bar{W}$ in Assumption 1.*

6. Experiments

In this section, we present empirical studies on both synthetic and real-world data sets. In the synthetic data experiments, we present the performance of the MSMTFL algorithm in terms of the parameter estimation error. In the real-world data experiments, we show the performance of the MSMTFL algorithm in terms of the prediction error.

6.1 Competing Algorithms

We present the empirical studies by comparing the following six algorithms:

- Lasso: ℓ_1-norm regularized feature learning algorithm with $\lambda\|W\|_{1,1}$ as the regularizer
- L1,2: $\ell_{1,2}$-norm regularized multi-task feature learning algorithm with $\lambda\|W\|_{1,2}$ as the regularizer (Obozinski et al., 2006)
- DirtyMTL: dirty model multi-task feature learning algorithm with $\lambda_s\|P\|_{1,1}+\lambda_b\|Q\|_{1,\infty}$ $(W=P+Q)$ as the regularizer (Jalali et al., 2010)
- CapL1,L1: our proposed multi-task feature learning algorithm with $\lambda\sum_{j=1}^{d}\min(\|\mathbf{w}^j\|_1,\theta)$ as the regularizer
- CapL1: capped-ℓ_1 regularized feature learning algorithm with $\lambda\sum_{j=1}^{d}\sum_{i=1}^{m}\min(|w_{ji}|,\theta)$ as the regularizer
- CapL1,L2: capped-ℓ_1,ℓ_2 regularized multi-task feature learning algorithm with $\lambda\sum_{j=1}^{d}\min(\|\mathbf{w}^j\|,\theta)$ as the regularizer

In the experiments, we employ the quadratic loss function in Equation (2) for all the compared algorithms. We use MSMTFL-type algorithms (similar to Algorithm 1) to sovle capped-ℓ_1 and capped-ℓ_1,ℓ_2 regularized feature learning problems (details are provided in Appendix C).

6.2 Synthetic Data Experiments

We generate synthetic data by setting the number of tasks as m and each task has n samples which are of dimensionality d; each element of the data matrix $X_i \in \mathbb{R}^{n\times d}$ $(i \in \mathbb{N}_m)$ for the i-th task is sampled i.i.d. from the Gaussian distribution $N(0,1)$ and we then normalize all columns to length 1; each entry of the underlying true weight $\bar{W} \in \mathbb{R}^{d\times m}$ is sampled i.i.d. from the uniform distribution in the interval $[-10,10]$; we randomly set 90% rows of $\bar{W}$ as zero vectors and 80% elements of the remaining nonzero entries as zeros; each entry of the noise $\boldsymbol{\delta}_i \in \mathbb{R}^n$ is sampled i.i.d. from the Gaussian distribution $N(0,\sigma^2)$; the responses are computed as $\mathbf{y}_i = X_i\bar{\mathbf{w}}_i + \boldsymbol{\delta}_i$ $(i \in \mathbb{N}_m)$.

We first report the averaged parameter estimation error $\|\hat{W} - \bar{W}\|_{2,1}$ vs. Stage (ℓ) plots for MSMTFL (Figure 1). We observe that the error decreases as ℓ increases, which shows the advantage of our proposed algorithm over Lasso. This is consistent with the theoretical result in Theorem 8. Moreover, the parameter estimation error decreases quickly and converges in a few stages.

We then report the averaged parameter estimation error $\|\hat{W} - \bar{W}\|_{2,1}$ in comparison with six algorithms in different parameter settings (Figure 2 and Figure 3). For a fair comparison, we compare the smallest estimation errors of the six algorithms in all the parameter settings as done in (Zhang, 2009, 2010). We observe that the parameter estimation errors of the capped-ℓ_1,ℓ_1, capped-ℓ_1 and capped-ℓ_1,ℓ_2 regularized feature learning formulations solved by MSMTFL-type algorithms are the smallest among all algorithms. In most cases, CapL1,L1 achieves a slightly smaller error than CapL1 and CapL1,L2. This empirical result demonstrates the effectiveness of the MSMTFL algorithm. We also have the following observations: (a) When λ is large enough, all six algorithms tend to have the same parameter estimation error. This is reasonable, because the solutions $\hat{W}$'s obtained by the six algorithms are all zero matrices, when λ is very large. (b) The performance of the MSMTFL algorithm is similar for different θ's, when λ exceeds a certain value.

6.3 Real-World Data Experiments

We conduct experiments on two real-world data sets: MRI and Isolet data sets.

The MRI data set is collected from the ANDI database, which contains 675 patients' MRI data preprocessed using FreeSurfer.[2]. The MRI data include 306 features and the response (target) is the Mini Mental State Examination (MMSE) score coming from 6 different time points: M06, M12, M18, M24, M36, and M48. We remove the samples which fail the MRI quality controls and have missing entries. Thus, we have 6 tasks with each task corresponding to a time point and the sample sizes corresponding to 6 tasks are 648, 642, 293, 569, 389 and 87, respectively.

The Isolet data set[3] is collected from 150 speakers who speak the name of each English letter of the alphabet twice. Thus, there are 52 samples from each speaker. The speakers are grouped into 5 subsets which respectively include 30 similar speakers, and the subsets are named Isolet1, Isolet2, Isolet3, Isolet4, and Isolet5. Thus, we naturally have 5 tasks with each task corresponding to one subset. The 5 tasks respectively have 1560, 1560, 1560, 1558, and 1559 samples,[4] where each sample includes 617 features and the response is the English letter label (1-26).

In the experiments, we treat the MMSE and letter labels as the regression values for the MRI data set and the Isolet data set, respectively. For both data sets, we randomly extract the training samples from each task with different training ratios (15%, 20% and 25%) and use the rest of samples to

2. FreeSurfer can be found at `www.loni.ucla.edu/ADNI/`.
3. The data set can be found at `www.zjucadcg.cn/dengcai/Data/data.html`.
4. Three samples are historically missing.

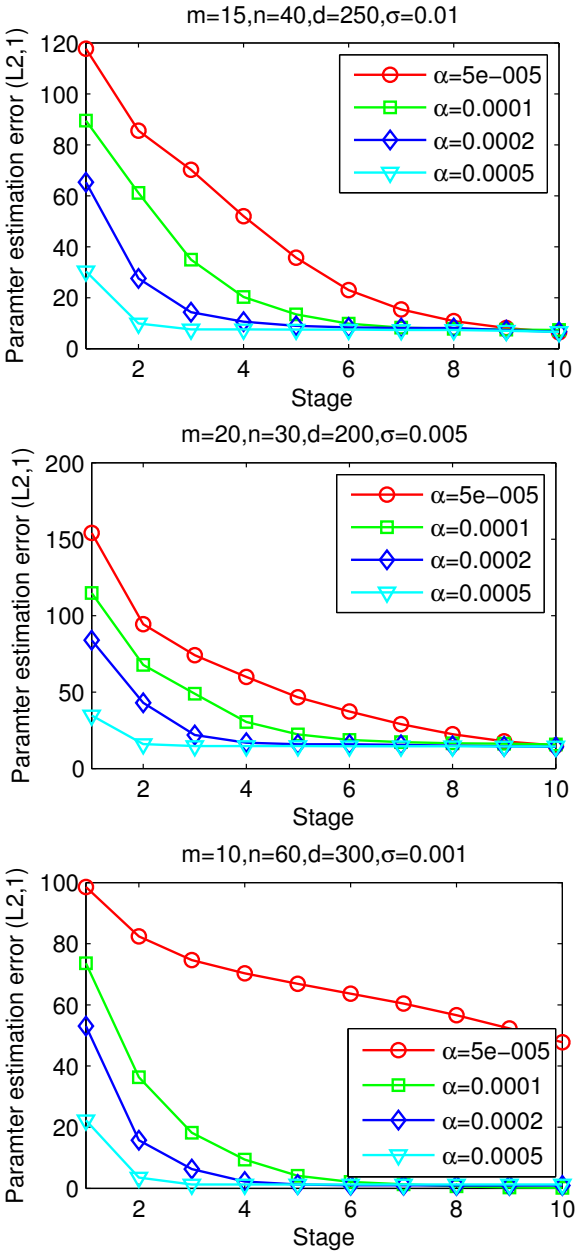


Figure 1: Averaged parameter estimation error $\|\hat{W} - \bar{W}\|_{2,1}$ vs. Stage (ℓ) plots for MSMTFL on the synthetic data set (averaged over 10 runs). Here we set $\lambda = \alpha\sqrt{\ln(dm)/n}$, $\theta = 50m\lambda$. Note that $\ell = 1$ corresponds to Lasso; the results show the stage-wise improvement over Lasso.

form the test set. We evaluate the six algorithms in terms of the normalized mean squared error (nMSE) and the averaged means squared error (aMSE), which are commonly used in multi-task learning problems (Zhang and Yeung, 2010; Zhou et al., 2011; Gong et al., 2012). For each training

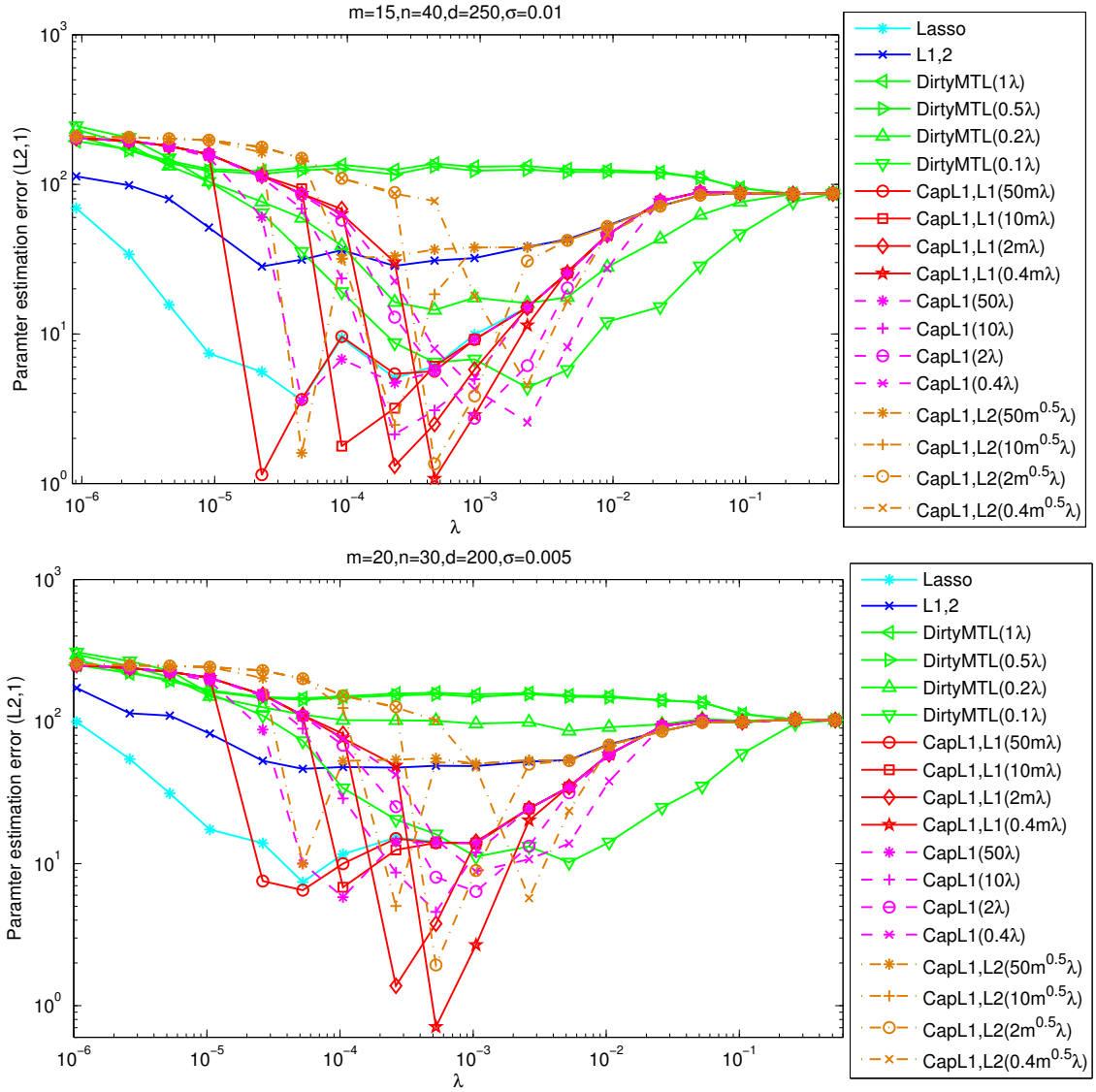


Figure 2: Averaged parameter estimation error $\|\hat{W} - \bar{W}\|_{2,1}$ vs. λ plots on the synthetic data set (averaged over 10 runs). DirtyMTL, CapL1,L1, CapL1, CapL1,L2 have two parameters; we set $\lambda_s/\lambda_b = 1, 0.5, 0.2, 0.1$ for DirtyMTL ($1/m \le \lambda_s/\lambda_b \le 1$ was adopted in Jalali et al. (2010)), $\theta/\lambda = 50m, 10m, 2m, 0.4m$ for CapL1,L1, $\theta/\lambda = 50, 10, 2, 0.4$ for CapL1 and $\theta/\lambda = 50m^{0.5}, 10m^{0.5}, 2m^{0.5}, 0.4m^{0.5}$ for CapL1,L2 (The settings of θ/λ for CapL1,L1, CapL1 and CapL1,L2 are based on the relationships of $\|\mathbf{w}^j\|_1, |w_{ji}|$ and $\|\mathbf{w}^j\|$, where $\mathbf{w}^j \in \mathbb{R}^{1\times m}$ and w_{ji} are the j-th row and the (j,i)-th entry of W, respectively).

ratio, both nMSE and aMSE are averaged over 10 random splittings of training and test sets and the standard deviation is also shown. All parameters of the six algorithms are tuned via 3-fold cross validation.

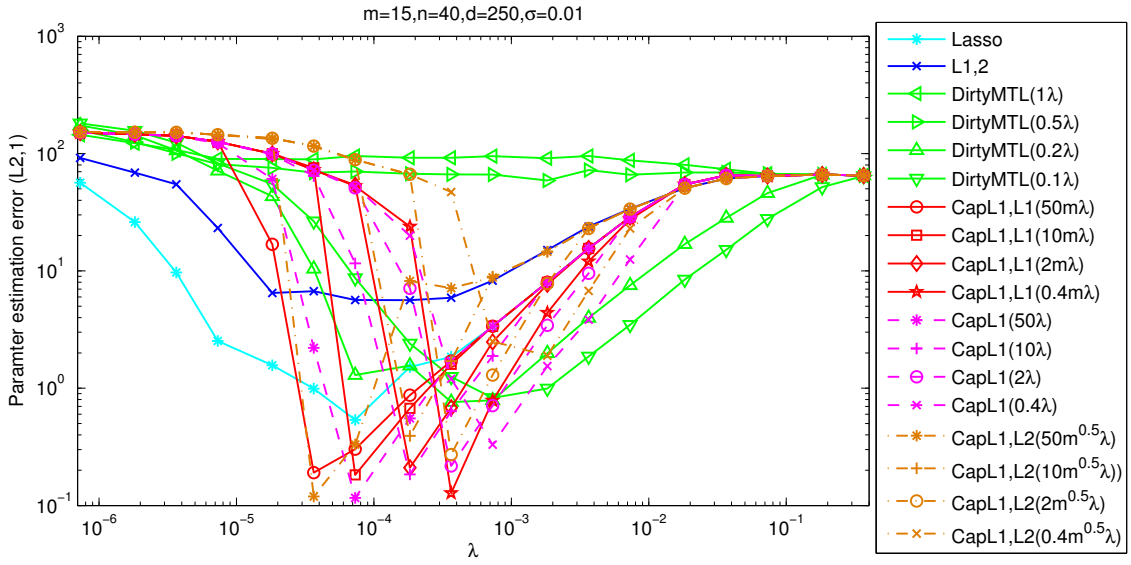


Figure 3: (continued) Averaged parameter estimation error $\|\hat{W}-\bar{W}\|_{2,1}$ vs. λ plots on the synthetic data set (averaged over 10 runs). DirtyMTL, CapL1,L1, CapL1, CapL1,L2 have two parameters; we set $\lambda_s/\lambda_b = 1, 0.5, 0.2, 0.1$ for DirtyMTL ($1/m \le \lambda_s/\lambda_b \le 1$ was adopted in Jalali et al. (2010)), $\theta/\lambda = 50m, 10m, 2m, 0.4m$ for CapL1,L1, $\theta/\lambda = 50, 10, 2, 0.4$ for CapL1 and $\theta/\lambda = 50m^{0.5}, 10m^{0.5}, 2m^{0.5}, 0.4m^{0.5}$ for CapL1,L2 (The settings of θ/λ for CapL1,L1, CapL1 and CapL1,L2 are based on the relationships of $\|\mathbf{w}^j\|_1, |w_{ji}|$ and $\|\mathbf{w}^j\|$, where $\mathbf{w}^j \in \mathbb{R}^{1\times m}$ and w_{ji} are the j-th row and the (j,i)-th entry of W, respectively).

Table 1 and Table 2 show the experimental results in terms of the averaged nMSE (aMSE) and the standard deviation. From these results, we observe that CapL1,L1 and CapL1,L2 outperform all the other competing feature learning algorithms on both data sets in terms of the regression errors (nMSE and aMSE). On the MRI data set, CapL1,L1 achieves slightly better performance than CapL1,L2 and on the Isolet data set, CapL1,L2 achieves slightly better performance than CapL1,L1. These empirical results demonstrate the effectiveness of the proposed MSMTFL (-type) algorithms.

7. Conclusions

In this paper, we propose a non-convex formulation for multi-task feature learning, which learns the specific features of each task as well as the common features shared among tasks. The non-convex formulation adopts the capped-ℓ_1,ℓ_1 regularizer to better approximate the ℓ_0-type one than the commonly used convex regularizer. To solve the non-convex optimization problem, we propose a Multi-Stage Multi-Task Feature Learning (MSMTFL) algorithm and provide intuitive interpretations from several aspects. We also present a detailed convergence analysis and discuss the reproducibility issue for the proposed algorithm. Specifically, we show that, under a mild condition, the solution generated by MSMTFL is unique. Although the solution may not be globally optimal, we theoretically show that it has good performance in terms of the parameter estimation error bound. Experimental results on both synthetic and real-world data sets demonstrate the effectiveness of our

measure	traning ratio	Lasso	L1,2	DirtyMTL
nMSE	0.15	0.6577(0.0193)	0.6443(0.0326)	0.6150(0.0160)
	0.20	0.6294(0.0255)	0.6541(0.0182)	0.6110(0.0122)
	0.25	0.6007(0.0120)	0.6407(0.0310)	0.5997(0.0218)
aMSE	0.15	0.0190(0.0008)	0.0184(0.0006)	0.0173(0.0006)
	0.20	0.0178(0.0009)	0.0184(0.0005)	0.0170(0.0007)
	0.25	0.0173(0.0007)	0.0183(0.0004)	0.0169(0.0007)
measure	traning ratio	CapL1,L1	CapL1	CapL1,L2
nMSE	0.15	**0.5551(0.0082)**	0.6448(0.0238)	**0.5591(0.0082)**
	0.20	**0.5539(0.0094)**	0.6245(0.0396)	**0.5612(0.0086)**
	0.25	**0.5513(0.0097)**	0.5899(0.0203)	**0.5595(0.0063)**
aMSE	0.15	**0.0163(0.0007)**	0.0187(0.0009)	**0.0165(0.0007)**
	0.20	**0.0161(0.0006)**	0.0177(0.0010)	**0.0163(0.0006)**
	0.25	**0.0162(0.0007)**	0.0171(0.0009)	**0.0164(0.0007)**

Table 1: Comparison of six feature learning algorithms on the MRI data set in terms of the averaged nMSE and aMSE (standard deviation), which are averaged over 10 random splittings. The two best results are in bold.

measure	traning ratio	Lasso	L1,2	DirtyMTL
nMSE	0.15	0.6798(0.0120)	0.6788(0.0149)	0.6427(0.0172)
	0.2	0.6465(0.0105)	0.6778(0.0104)	0.6371(0.0111)
	0.25	0.6279(0.0099)	0.6666(0.0110)	0.6304(0.0093)
aMSE	0.15	0.1605(0.0028)	0.1602(0.0033)	0.1517(0.0039)
	0.2	0.1522(0.0022)	0.1596(0.0021)	0.1500(0.0023)
	0.25	0.1477(0.0024)	0.1568(0.0025)	0.1482(0.0019)
measure	traning ratio	CapL1,L1	CapL1	CapL1,L2
nMSE	0.15	**0.6421(0.0153)**	0.6541(0.0122)	**0.5819(0.0125)**
	0.2	**0.5847(0.0081)**	0.5962(0.0051)	**0.5589(0.0056)**
	0.25	**0.5496(0.0106)**	0.5569(0.0158)	**0.5422(0.0063)**
aMSE	0.15	**0.1516(0.0035)**	0.1544(0.0028)	**0.1373(0.0030)**
	0.2	**0.1376(0.0020)**	0.1404(0.0012)	**0.1316(0.0014)**
	0.25	**0.1293(0.0028)**	0.1310(0.0042)	**0.1275(0.0013)**

Table 2: Comparison of six feature learning algorithms on the Isolet data set in terms of the averaged nMSE and aMSE (standard deviation), which are averaged over 10 random splittings. The two best results are in bold.

proposed MSMTFL algorithm in comparison with the state of the art multi-task feature learning algorithms.

There are several interesting issues that need to be addressed in the future. First, we will explore the conditions under which a globally optimal solution of the proposed formulation can be obtained by the MSMTFL algorithm. Second, we plan to explore general theoretical bounds for multi-task learning settings (involving different loss functions and non-convex regularization terms) using multi-stage algorithms. Third, we will adapt the GIST algorithm (Gong et al., 2013a,b) to solve the non-convex multi-task feature learning problem and derive theoretical bounds.

Acknowledgments

We would like to thank the action editor and anonymous reviewers for their constructive comments. This work is partly supported by 973 Program (2013CB329503), NSFC (Grant No. 91120301, 61075004 and 61021063), NIH (R01 LM010730) and NSF (IIS-0953662, CCF-1025177).

Appendix A. Proofs of Lemmas 13 to 16

In this appendix, we provide detailed proofs for Lemmas 13 to 16. In our proofs, we use several lemmas (summarized in Appendix B) from Zhang (2010).

We first introduce some notations used in the proof. Define

$$\pi_i(k_i,s_i) = \sup_{\mathbf{v}\in\mathbb{R}^{k_i},\mathbf{u}\in\mathbb{R}^{s_i},I_i,\mathcal{J}_i} \frac{\mathbf{v}^T A^{(i)}_{I_i,\mathcal{J}_i}\mathbf{u}\|\mathbf{v}\|}{\mathbf{v}^T A^{(i)}_{I_i,I_i}\mathbf{v}\|\mathbf{u}\|_\infty},$$

where $s_i+k_i\le d$ with $s_i,k_i\ge 1$; I_i and $\mathcal{J}_i$ are *disjoint* subsets of $\mathbb{N}_d$ with k_i and s_i elements respectively (with some abuse of notation, we also let I_i be a subset of $\mathbb{N}_d\times\{i\}$, depending on the context.); $A^{(i)}_{I_i,\mathcal{J}_i}$ is a sub-matrix of $A_i = n^{-1}X_i^TX_i\in\mathbb{R}^{d\times d}$ with rows indexed by I_i and columns indexed by $\mathcal{J}_i$.

We let $\mathbf{w}_{I_i}$ be a $d\times 1$ vector with the j-th entry being w_{ji}, if $(j,i)\in I_i$, and 0, otherwise. We also let W_I be a $d\times m$ matrix with (j,i)-th entry being w_{ji}, if $(j,i)\in I$, and 0, otherwise.

Proof of Lemma 13 For the j-th entry of $\bar{\boldsymbol{\epsilon}}_i$ ($j\in\mathbb{N}_d$):

$$|\bar{\varepsilon}_{ji}| = \frac{1}{n}\left|\left(\mathbf{x}_j^{(i)}\right)^T(X_i\bar{\mathbf{w}}_i-\mathbf{y}_i)\right| = \frac{1}{n}\left|\left(\mathbf{x}_j^{(i)}\right)^T\boldsymbol{\delta}_i\right|,$$

where $\mathbf{x}_j^{(i)}$ is the j-th column of X_i. We know that the entries of $\boldsymbol{\delta}_i$ are independent sub-Gaussian random variables, and $\|1/n\mathbf{x}_j^{(i)}\|^2 = \|\mathbf{x}_j^{(i)}\|^2/n^2\le\rho_i^+(1)/n$. According to Lemma 18, we have $\forall t>0$:

$$\Pr(|\bar{\varepsilon}_{ji}|\ge t)\le 2\exp(-nt^2/(2\sigma^2\rho_i^+(1)))\le 2\exp(-nt^2/(2\sigma^2\rho^+_{max}(1))).$$

Thus we obtain:

$$\Pr(\|\bar{\Upsilon}\|_{\infty,\infty}\le t)\ge 1-2dm\exp(-nt^2/(2\sigma^2\rho^+_{max}(1))).$$

Let $\eta = 2dm\exp(-nt^2/(2\sigma^2\rho^+_{max}(1)))$ and we can obtain Equation (27). Equation (28) directly follows from Lemma 21 and the following fact:

$$\|\mathbf{x}_i\|^2\le ay_i\Rightarrow\|X\|_F^2=\sum_{i=1}^m\|\mathbf{x}_i\|^2\le ma\max_{i\in\mathbb{N}_m}y_i.$$

■

Proof of Lemma 14 The optimality condition of Equation (5) implies that

$$\frac{2}{n}X_i^T(X_i\hat{\mathbf{w}}_i-\mathbf{y}_i)+\hat{\boldsymbol{\lambda}}_i\odot\operatorname{sign}(\hat{\mathbf{w}}_i)=\mathbf{0},$$

where $\odot$ denotes the element-wise product; $\operatorname{sign}(\mathbf{w})=[\operatorname{sign}(w_1),\cdots,\operatorname{sign}(w_d)]^T$, where $\operatorname{sign}(w_i)=1$, if $w_i>0$; $\operatorname{sign}(w_i)=-1$, if $w_i<0$; and $\operatorname{sign}(w_i)\in[-1,1]$, otherwise. We note that $X_i\hat{\mathbf{w}}_i-\mathbf{y}_i=X_i\hat{\mathbf{w}}_i-X_i\bar{\mathbf{w}}_i+X_i\bar{\mathbf{w}}_i-\mathbf{y}_i$ and we can rewrite the above equation into the following form:

$$2A_i\Delta\hat{\mathbf{w}}_i=-2\bar{\boldsymbol{\epsilon}}_i-\hat{\boldsymbol{\lambda}}_i\odot\operatorname{sign}(\hat{\mathbf{w}}_i).$$

Thus, for all $\mathbf{v}\in\mathbb{R}^d$, we have

$$2\mathbf{v}^TA_i\Delta\hat{\mathbf{w}}_i=-2\mathbf{v}^T\bar{\boldsymbol{\epsilon}}_i-\sum_{j=1}^d\hat{\lambda}_{ji}v_j\operatorname{sign}(\hat{w}_{ji}). \tag{33}$$

Letting $\mathbf{v}=\Delta\hat{\mathbf{w}}_i$ and noticing that $\Delta\hat{w}_{ji}=\hat{w}_{ji}$ for $(j,i)\notin\bar{\mathcal{F}}_i, i\in\mathbb{N}_m$, we obtain

$$\begin{aligned}
0\le 2\Delta\hat{\mathbf{w}}_i^TA_i\Delta\hat{\mathbf{w}}_i&=-2\Delta\hat{\mathbf{w}}_i^T\bar{\boldsymbol{\epsilon}}_i-\sum_{j=1}^d\hat{\lambda}_{ji}\Delta\hat{w}_{ji}\operatorname{sign}(\hat{w}_{ji})\\
&\le 2\|\Delta\hat{\mathbf{w}}_i\|_1\|\bar{\boldsymbol{\epsilon}}_i\|_\infty-\sum_{(j,i)\in\bar{\mathcal{F}}_i}\hat{\lambda}_{ji}\Delta\hat{w}_{ji}\operatorname{sign}(\hat{w}_{ji})-\sum_{(j,i)\notin\bar{\mathcal{F}}_i}\hat{\lambda}_{ji}\Delta\hat{w}_{ji}\operatorname{sign}(\hat{w}_{ji})\\
&\le 2\|\Delta\hat{\mathbf{w}}_i\|_1\|\bar{\boldsymbol{\epsilon}}_i\|_\infty+\sum_{(j,i)\in\bar{\mathcal{F}}_i}\hat{\lambda}_{ji}|\Delta\hat{w}_{ji}|-\sum_{(j,i)\notin\bar{\mathcal{F}}_i}\hat{\lambda}_{ji}|\hat{w}_{ji}|\\
&\le 2\|\Delta\hat{\mathbf{w}}_i\|_1\|\bar{\boldsymbol{\epsilon}}_i\|_\infty+\sum_{(j,i)\in\bar{\mathcal{F}}_i}\hat{\lambda}_{ji}|\Delta\hat{w}_{ji}|-\sum_{(j,i)\in\mathcal{G}_i}\hat{\lambda}_{ji}|\hat{w}_{ji}|\\
&\le 2\|\Delta\hat{\mathbf{w}}_i\|_1\|\bar{\boldsymbol{\epsilon}}_i\|_\infty+\sum_{(j,i)\in\bar{\mathcal{F}}_i}\hat{\lambda}_{0i}|\Delta\hat{w}_{ji}|-\sum_{(j,i)\in\mathcal{G}_i}\hat{\lambda}_{\mathcal{G}_i}|\hat{w}_{ji}|\\
&=\sum_{(j,i)\in\mathcal{G}_i}(2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty-\hat{\lambda}_{\mathcal{G}_i})|\hat{w}_{ji}|+\sum_{(j,i)\notin\bar{\mathcal{F}}_i\cup\mathcal{G}_i}2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty|\hat{w}_{ji}|+\sum_{(j,i)\in\bar{\mathcal{F}}_i}(2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty+\hat{\lambda}_{0i})|\Delta\hat{w}_{ji}|.
\end{aligned}$$

The last equality above is due to $\mathbb{N}_d\times\{i\}=\mathcal{G}_i\cup(\bar{\mathcal{F}}_i\cup\mathcal{G}_i)^c\cup\bar{\mathcal{F}}_i$ and $\Delta\hat{w}_{ji}=\hat{w}_{ji},\forall(j,i)\notin\bar{\mathcal{F}}_i\supseteq\mathcal{G}_i$. Rearranging the above inequality and noticing that $2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty<\hat{\lambda}_{\mathcal{G}_i}\le\hat{\lambda}_{0i}$, we obtain:

$$\begin{aligned}
\sum_{(j,i)\in\mathcal{G}_i}|\hat{w}_{ji}|&\le\frac{2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}{\hat{\lambda}_{\mathcal{G}_i}-2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}\sum_{(j,i)\notin\bar{\mathcal{F}}_i\cup\mathcal{G}_i}|\hat{w}_{ji}|+\frac{2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty+\hat{\lambda}_{0i}}{\hat{\lambda}_{\mathcal{G}_i}-2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}\sum_{(j,i)\in\bar{\mathcal{F}}_i}|\Delta\hat{w}_{ji}|\\
&\le\frac{2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty+\hat{\lambda}_{0i}}{\hat{\lambda}_{\mathcal{G}_i}-2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}\|\Delta\hat{\mathbf{w}}_{\mathcal{G}_i^c}\|_1.
\end{aligned}\tag{34}$$

Then Lemma 14 can be obtained from the above inequality and the following two inequalities.

$$\max_{i\in\mathbb{N}_m}\frac{2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty+\hat{\lambda}_{0i}}{\hat{\lambda}_{\mathcal{G}_i}-2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}\le\frac{2\|\bar{\Upsilon}\|_{\infty,\infty}+\hat{\lambda}_0}{\hat{\lambda}_{\mathcal{G}}-2\|\bar{\Upsilon}\|_{\infty,\infty}}\text{ and }\sum_{i=1}^m x_iy_i\le\|\mathbf{x}\|_\infty\|\mathbf{y}\|_1.$$

■

Proof of Lemma 15 According to the definition of $\mathcal{G}$ ($\mathcal{G}_{(\ell)}$), we know that $\bar{\mathcal{F}}_i \cap \mathcal{G}_i = \emptyset$ ($i \in \mathbb{N}_m$) and $\forall (j,i) \in \mathcal{G}$ ($\mathcal{G}_{(\ell)}$), $\hat{\lambda}_{ji}^{(\ell-1)} = \lambda$. Thus, all conditions of Lemma 14 are satisfied, by noticing the relationship between Equation (22) and Equation (27). Based on the definition of $\mathcal{G}$ ($\mathcal{G}_{(\ell)}$), we easily obtain $\forall j \in \mathbb{N}_d$:

$$(j,i) \in \mathcal{G}_i, \forall i \in \mathbb{N}_m \text{ or } (j,i) \notin \mathcal{G}_i, \forall i \in \mathbb{N}_m.$$

and hence $k_\ell = |\mathcal{G}_1^c| = \cdots = |\mathcal{G}_m^c|$ (k_ℓ is some integer). Now, we assume that at stage $\ell \geq 1$:

$$k_\ell = |\mathcal{G}_1^c| = \cdots = |\mathcal{G}_m^c| \leq 2\bar{r}. \tag{35}$$

We will show in the second part of this proof that Equation (35) holds for all ℓ. Based on Lemma 19 and Equation (21), we have:

$$\begin{aligned}
\pi_i(2\bar{r}+s,s) &\leq \frac{s^{1/2}}{2}\sqrt{\rho_i^+(s)/\rho_i^-(2\bar{r}+2s)-1} \\
&\leq \frac{s^{1/2}}{2}\sqrt{1+s/(2\bar{r})-1} \\
&= 0.5s(2\bar{r})^{-1/2},
\end{aligned}$$

which indicates that

$$0.5 \leq t_i = 1 - \pi_i(2\bar{r}+s,s)(2\bar{r})^{1/2}s^{-1} \leq 1.$$

For all $t_i \in [0.5,1]$, under the conditions of Equation (22) and Equation (27), we have

$$\frac{2\|\bar{\epsilon}_i\|_\infty + \lambda}{\lambda - 2\|\bar{\epsilon}_i\|_\infty} \leq \frac{2\|\bar{\Upsilon}\|_{\infty,\infty} + \lambda}{\lambda - 2\|\bar{\Upsilon}\|_{\infty,\infty}} \leq \frac{7}{5} \leq \frac{4-t_i}{4-3t_i} \leq 3.$$

Following Lemma 14, we have

$$\|\hat{W}_{\mathcal{G}}\|_{1,1} \leq 3\|\Delta\hat{W}_{\mathcal{G}^c}\|_{1,1} = 3\|\Delta\hat{W} - \Delta\hat{W}_{\mathcal{G}}\|_{1,1} = 3\|\Delta\hat{W} - \hat{W}_{\mathcal{G}}\|_{1,1}.$$

Therefore

$$\begin{aligned}
&\|\Delta\hat{W} - \Delta\hat{W}_I\|_{\infty,1} = \|\Delta\hat{W}_{\mathcal{G}} - \Delta\hat{W}_{\mathcal{J}}\|_{\infty,1} \\
&\leq \|\Delta\hat{W}_{\mathcal{J}}\|_{1,1}/s = (\|\Delta\hat{W}_{\mathcal{G}}\|_{1,1} - \|\Delta\hat{W} - \Delta\hat{W}_I\|_{1,1})/s \\
&\leq s^{-1}(3\|\Delta\hat{W} - \hat{W}_{\mathcal{G}}\|_{1,1} - \|\Delta\hat{W} - \Delta\hat{W}_I\|_{1,1}),
\end{aligned}$$

which implies that

$$\begin{aligned}
&\|\Delta\hat{W}\|_{2,1} - \|\Delta\hat{W}_I\|_{2,1} \leq \|\Delta\hat{W} - \Delta\hat{W}_I\|_{2,1} \\
&\leq (\|\Delta\hat{W} - \Delta\hat{W}_I\|_{1,1}\|\Delta\hat{W} - \Delta\hat{W}_I\|_{\infty,1})^{1/2} \\
&\leq \left(\|\Delta\hat{W} - \Delta\hat{W}_I\|_{1,1}\right)^{1/2}\left(s^{-1}(3\|\Delta\hat{W} - \hat{W}_{\mathcal{G}}\|_{1,1} - \|\Delta\hat{W} - \Delta\hat{W}_I\|_{1,1})\right)^{1/2} \\
&\leq \left(\left(3\|\Delta\hat{W} - \hat{W}_{\mathcal{G}}\|_{1,1}/2\right)^2\right)^{1/2} s^{-1/2} \\
&\leq (3/2)s^{-1/2}(2\bar{r})^{1/2}\|\Delta\hat{W} - \hat{W}_{\mathcal{G}}\|_{2,1} \\
&\leq (3/2)(2\bar{r}/s)^{1/2}\|\Delta\hat{W}_I\|_{2,1}.
\end{aligned}$$

In the above derivation, the third inequality is due to $a(3b-a) \leq (3b/2)^2$, and the fourth inequality follows from Equation (35) and $\bar{\mathcal{F}} \cap \mathcal{G} = \emptyset \Rightarrow \Delta\hat{W}_{\mathcal{G}} = \hat{W}_{\mathcal{G}}$. Rearranging the above inequality, we obtain at stage ℓ:

$$\|\Delta\hat{W}\|_{2,1} \leq \left(1 + 1.5\sqrt{\frac{2\bar{r}}{s}}\right) \|\Delta\hat{W}_I\|_{2,1}. \tag{36}$$

From Lemma 20, we have:

$$\begin{aligned}
&\max(0, \Delta\hat{\mathbf{w}}_{I_i}^T A_i \Delta\hat{\mathbf{w}}_i) \\
&\geq \rho_i^-(k_\ell + s)(\|\Delta\hat{\mathbf{w}}_{I_i}\| - \pi_i(k_\ell + s, s)\|\hat{\mathbf{w}}_{\mathcal{G}_i}\|_1/s)\|\Delta\hat{\mathbf{w}}_{I_i}\| \\
&\geq \rho_i^-(k_\ell + s)[1 - (1-t_i)(4-t_i)/(4-3t_i)]\|\Delta\hat{\mathbf{w}}_{I_i}\|^2 \\
&\geq 0.5 t_i \rho_i^-(k_\ell + s)\|\Delta\hat{\mathbf{w}}_{I_i}\|^2 \\
&\geq 0.25\rho_i^-(2\bar{r} + s)\|\Delta\hat{\mathbf{w}}_{I_i}\|^2 \\
&\geq 0.25\rho_{min}^-(2\bar{r} + s)\|\Delta\hat{\mathbf{w}}_{I_i}\|^2,
\end{aligned}$$

where the second inequality is due to Equation (34), that is

$$\begin{aligned}
\|\hat{\mathbf{w}}_{\mathcal{G}_i}\|_1 &\leq \frac{2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty + \hat{\lambda}_{0i}}{\hat{\lambda}_{\mathcal{G}_i} - 2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}\|\Delta\hat{\mathbf{w}}_{\mathcal{G}_i^c}\|_1 \\
&\leq \frac{(2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty + \hat{\lambda}_{0i})\sqrt{k_\ell}}{\hat{\lambda}_{\mathcal{G}_i} - 2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}\|\Delta\hat{\mathbf{w}}_{\mathcal{G}_i^c}\| \\
&\leq \frac{(2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty + \hat{\lambda}_{0i})\sqrt{k_\ell}}{\hat{\lambda}_{\mathcal{G}_i} - 2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty}\|\Delta\hat{\mathbf{w}}_{I_i}\| \\
&\leq \frac{(4-t_i)\sqrt{k_\ell}}{4-3t_i}\|\Delta\hat{\mathbf{w}}_{I_i}\|;
\end{aligned}$$

the third inequality follows from $1 - (1-t_i)(4-t_i)/(4-3t_i) \geq 0.5t_i$ for $t_i \in [0.5, 1]$ and the fourth inequality follows from the assumption in Equation (35) and $t_i \geq 0.5$.

If $\Delta\hat{\mathbf{w}}_{I_i}^T A_i \Delta\hat{\mathbf{w}}_i \leq 0$, then $\|\Delta\hat{\mathbf{w}}_{I_i}\| = 0$. If $\Delta\hat{\mathbf{w}}_{I_i}^T A_i \Delta\hat{\mathbf{w}}_i > 0$, then we have

$$\Delta\hat{\mathbf{w}}_{I_i}^T A_i \Delta\hat{\mathbf{w}}_i \geq 0.25\rho_{min}^-(2\bar{r} + s)\|\Delta\hat{\mathbf{w}}_{I_i}\|^2. \tag{37}$$

By letting $\mathbf{v} = \Delta\hat{\mathbf{w}}_{I_i}$, we obtain the following from Equation (33):

$$
\begin{aligned}
2\Delta\hat{\mathbf{w}}_{I_i}^T A_i \Delta\hat{\mathbf{w}}_i &= -2\Delta\hat{\mathbf{w}}_{I_i}^T \bar{\boldsymbol{\epsilon}}_i - \sum_{(j,i)\in I_i} \hat{\lambda}_{ji}\Delta\hat{w}_{ji}\text{sign}(\hat{w}_{ji}) \\
&= -2\Delta\hat{\mathbf{w}}_{I_i}^T \bar{\boldsymbol{\epsilon}}_{\mathcal{G}_i^c} - 2\Delta\hat{\mathbf{w}}_{I_i}^T \bar{\boldsymbol{\epsilon}}_{\mathcal{G}_i} - \sum_{(j,i)\in\bar{\mathcal{F}}_i} \hat{\lambda}_{ji}\Delta\hat{w}_{ji}\text{sign}(\hat{w}_{ji}) \\
&\quad - \sum_{(j,i)\in\mathcal{J}_i} \hat{\lambda}_{ji}|\Delta\hat{w}_{ji}| - \sum_{(j,i)\in\bar{\mathcal{F}}_i^c\cap\mathcal{G}_i^c} \hat{\lambda}_{ji}|\Delta\hat{w}_{ji}| \\
&= -2\Delta\hat{\mathbf{w}}_{I_i}^T \bar{\boldsymbol{\epsilon}}_{\mathcal{G}_i^c} - 2\Delta\hat{\mathbf{w}}_{\mathcal{J}_i}^T \bar{\boldsymbol{\epsilon}}_{\mathcal{J}_i} - \sum_{(j,i)\in\bar{\mathcal{F}}_i} \hat{\lambda}_{ji}\Delta\hat{w}_{ji}\text{sign}(\hat{w}_{ji}) \\
&\quad - \sum_{(j,i)\in\mathcal{J}_i} \hat{\lambda}_{ji}|\Delta\hat{w}_{ji}| - \sum_{(j,i)\in\bar{\mathcal{F}}_i^c\cap\mathcal{G}_i^c} \hat{\lambda}_{ji}|\Delta\hat{w}_{ji}| \\
&\le 2\|\Delta\hat{\mathbf{w}}_{I_i}\|\|\bar{\boldsymbol{\epsilon}}_{\mathcal{G}_i^c}\| + 2\|\bar{\boldsymbol{\epsilon}}_{\mathcal{J}_i}\|_\infty \sum_{(j,i)\in\mathcal{J}_i} |\Delta\hat{w}_{ji}| + \sum_{(j,i)\in\bar{\mathcal{F}}_i} \hat{\lambda}_{ji}|\Delta\hat{w}_{ji}| - \sum_{(j,i)\in\mathcal{J}_i} \hat{\lambda}_{ji}|\Delta\hat{w}_{ji}| \\
&\le 2\|\Delta\hat{\mathbf{w}}_{I_i}\|\|\bar{\boldsymbol{\epsilon}}_{\mathcal{G}_i^c}\| + \left(\sum_{(j,i)\in\bar{\mathcal{F}}_i} \hat{\lambda}_{ji}^2\right)^{1/2} \|\Delta\hat{\mathbf{w}}_{\bar{\mathcal{F}}_i}\| \\
&\le 2\|\Delta\hat{\mathbf{w}}_{I_i}\|\|\bar{\boldsymbol{\epsilon}}_{\mathcal{G}_i^c}\| + \left(\sum_{(j,i)\in\bar{\mathcal{F}}_i} \hat{\lambda}_{ji}^2\right)^{1/2} \|\Delta\hat{\mathbf{w}}_{I_i}\|. \qquad (38)
\end{aligned}
$$

In the above derivation, the second equality is due to $I_i = \mathcal{J}_i \cup \bar{\mathcal{F}}_i \cup (\bar{\mathcal{F}}_i^c \cap \mathcal{G}_i^c)$; the third equality is due to $I_i \cap \mathcal{G}_i = \mathcal{J}_i$; the second inequality follows from $\forall (j,i) \in \mathcal{J}_i, \hat{\lambda}_{ji} = \lambda \ge 2\|\bar{\boldsymbol{\epsilon}}_i\|_\infty \ge 2\|\bar{\boldsymbol{\epsilon}}_{\mathcal{J}_i}\|_\infty$ and the last inequality follows from $\bar{\mathcal{F}}_i \subseteq \mathcal{G}_i^c \subseteq I_i$. Combining Equation (37) and Equation (38), we have

$$
\|\Delta\hat{\mathbf{w}}_{I_i}\| \le \frac{2}{\rho_{min}^-(2\bar{r}+s)} \left[2\|\bar{\boldsymbol{\epsilon}}_{\mathcal{G}_i^c}\| + \left(\sum_{(j,i)\in\bar{\mathcal{F}}_i} \hat{\lambda}_{ji}^2\right)^{1/2} \right].
$$

Notice that

$$
\|\mathbf{x}_i\| \le a(\|\mathbf{y}_i\| + \|\mathbf{z}_i\|) \Rightarrow \|X\|_{2,1}^2 \le m\|X\|_F^2 = m\sum_i \|\mathbf{x}_i\|^2 \le 2ma^2(\|Y\|_F^2 + \|Z\|_F^2).
$$

Thus, we have

$$
\|\Delta\hat{W}_I\|_{2,1} \le \frac{\sqrt{8m\left(4\|\bar{\Upsilon}_{\mathcal{G}_{(\ell)}^c}\|_F^2 + \sum_{(j,i)\in\bar{\mathcal{F}}} (\hat{\lambda}_{ji}^{(\ell-1)})^2\right)}}{\rho_{min}^-(2\bar{r}+s)}. \qquad (39)
$$

Therefore, at stage ℓ, Equation (29) in Lemma 15 directly follows from Equation (36) and Equation (39). Following Equation (29), we have:

$$\begin{aligned}
&\|\hat{W}^{(\ell)} - \bar{W}\|_{2,1} = \|\Delta\hat{W}^{(\ell)}\|_{2,1} \\
&\leq \frac{\left(1+1.5\sqrt{\frac{2\bar{r}}{s}}\right)\sqrt{8m\left(4\|\bar{\Upsilon}_{\mathcal{G}^c_{(\ell)}}\|_F^2 + \sum_{(j,i)\in\bar{\mathcal{F}}}(\hat{\lambda}_{ji}^{(\ell-1)})^2\right)}}{\rho^-_{min}(2\bar{r}+s)} \\
&\leq \frac{8.83\sqrt{m}\sqrt{4\|\Upsilon\|^2_{\infty,\infty}|\mathcal{G}^c_{(\ell)}| + \bar{r}m\lambda^2}}{\rho^-_{min}(2\bar{r}+s)} \\
&\leq \frac{8.83\sqrt{m}\lambda\sqrt{\frac{8}{144}\bar{r}m + \bar{r}m}}{\rho^-_{min}(2\bar{r}+s)} \\
&\leq \frac{9.1m\lambda\sqrt{\bar{r}}}{\rho^-_{min}(2\bar{r}+s)},
\end{aligned}$$

where the first inequality is due to Equation (39); the second inequality is due to $s \geq \bar{r}$ (assumption in Theorem 8), $\hat{\lambda}_{ji} \leq \lambda$, $\bar{r}m = |\bar{\mathcal{H}}| \geq |\bar{\mathcal{F}}|$ and the third inequality follows from Equation (35) and $\|\bar{\Upsilon}\|^2_{\infty,\infty} \leq (1/144)\lambda^2$. Therefore, Equation (30) in Lemma 15 holds at stage ℓ.

Notice that we obtain Lemma 15 at stage ℓ, by assuming that Equation (35) is satisfied. To prove that Lemma 15 holds for all stages, we next need to prove by induction that Equation (35) holds at all stages.

When $\ell = 1$, we have $\mathcal{G}^c_{(1)} = \bar{\mathcal{H}}$, which implies that Equation (35) holds. Now, we assume that Equation (35) holds at stage ℓ. Thus, by hypothesis induction, we have:

$$\begin{aligned}
\sqrt{|\mathcal{G}^c_{(\ell+1)} \setminus \bar{\mathcal{H}}|} &\leq \sqrt{m\theta^{-2}\|\hat{W}^{(\ell)}_{\mathcal{G}^c_{(\ell+1)}\setminus\bar{\mathcal{H}}} - \bar{W}_{\mathcal{G}^c_{(\ell+1)}\setminus\bar{\mathcal{H}}}\|^2_{2,1}} \\
&\leq \sqrt{m}\theta^{-1}\left\|\hat{W}^{(\ell)} - \bar{W}\right\|_{2,1} \\
&\leq \frac{9.1m^{3/2}\lambda\sqrt{\bar{r}}\theta^{-1}}{\rho^-_{min}(2\bar{r}+s)} \\
&\leq \sqrt{\bar{r}m},
\end{aligned}$$

where θ is the thresholding parameter in Equation (1); the first inequality above follows from the definition of $\mathcal{G}_{(\ell)}$ in Lemma 15:

$$\begin{aligned}
&\forall (j,i) \in \mathcal{G}^c_{(\ell+1)} \setminus \bar{\mathcal{H}}, \|(\hat{\mathbf{w}}^{(\ell)})^j\|_1^2/\theta^2 = \|(\hat{\mathbf{w}}^{(\ell)})^j - \bar{\mathbf{w}}^j\|_1^2/\theta^2 \geq 1 \\
\Rightarrow &|\mathcal{G}^c_{(\ell+1)} \setminus \bar{\mathcal{H}}| \leq m\theta^{-2}\|\hat{W}^{(\ell)}_{\mathcal{G}^c_{(\ell+1)}\setminus\bar{\mathcal{H}}} - \bar{W}_{\mathcal{G}^c_{(\ell+1)}\setminus\bar{\mathcal{H}}}\|^2_{2,1};
\end{aligned}$$

the last inequality is due to Equation (23). Thus, we have:

$$|\mathcal{G}^c_{(\ell+1)} \setminus \bar{\mathcal{H}}| \leq \bar{r}m \Rightarrow |\mathcal{G}^c_{(\ell+1)}| \leq 2\bar{r}m \Rightarrow k_{\ell+1} \leq 2\bar{r}.$$

Therefore, Equation (35) holds at all stages. Thus the two inequalities in Lemma 15 hold at all stages. This completes the proof of the lemma. ■

Proof of Lemma 16 The first inequality directly follows from $\bar{\mathcal{H}} \supseteq \bar{\mathcal{F}}$. Next, we focus on the second inequality. For each $(j,i) \in \bar{\mathcal{F}}$ $(\bar{\mathcal{H}})$, if $\|\hat{\mathbf{w}}^j\|_1 < \theta$, by considering Equation (20), we have

$$\|\bar{\mathbf{w}}^j - \hat{\mathbf{w}}^j\|_1 \geq \|\bar{\mathbf{w}}^j\|_1 - \|\hat{\mathbf{w}}^j\|_1 \geq 2\theta - \theta = \theta.$$

Therefore, we have for each $(j,i) \in \bar{\mathcal{F}}$ $(\bar{\mathcal{H}})$:

$$I\left(\|\hat{\mathbf{w}}^j\|_1 < \theta\right) \leq \|\bar{\mathbf{w}}^j - \hat{\mathbf{w}}^j\|_1/\theta.$$

Thus, the second inequality of Lemma 16 directly follows from the above inequality. ■

Appendix B. Lemmas from Zhang (2010)

Lemma 18 *Let $\mathbf{a} \in \mathbb{R}^n$ be a fixed vector and $\mathbf{x} \in \mathbb{R}^n$ be a random vector which is composed of independent sub-Gaussian components with parameter σ. Then we have:*

$$\Pr(|\mathbf{a}^T\mathbf{x}| \geq t) \leq 2\exp\left(-t^2/(2\sigma^2\|\mathbf{a}\|^2)\right), \forall t > 0.$$

Lemma 19 *The following inequality holds:*

$$\pi_i(k_i, s_i) \leq \frac{s_i^{1/2}}{2}\sqrt{\rho_i^+(s_i)/\rho_i^-(k_i+s_i) - 1}.$$

Lemma 20 *Let $\mathcal{G}_i \subseteq \mathbb{N}_d \times \{i\}$ such that $|\mathcal{G}_i^c| = k_i$, and let $\mathcal{J}_i$ be indices of the s_i largest components (in absolute values) of $\mathbf{w}_{\mathcal{G}_i}$ and $I_i = \mathcal{G}_i^c \cup \mathcal{J}_i$. Then for any $\mathbf{w}_i \in \mathbb{R}^d$, we have*

$$\max(0, \mathbf{w}_{I_i}^T A_i \mathbf{w}_i) \geq \rho_i^-(k_i+s_i)(\|\mathbf{w}_{I_i}\| - \pi_i(k_i+s_i, s_i)\|\mathbf{w}_{\mathcal{G}_i}\|_1/s_i)\|\mathbf{w}_{I_i}\|.$$

Lemma 21 *Let $\bar{\epsilon}_i = [\bar{\epsilon}_{1i}, \cdots, \bar{\epsilon}_{di}] = \frac{1}{n}X_i^T(X_i\bar{\mathbf{w}}_i - \mathbf{y}_i)$ $(i \in \mathbb{N}_m)$, and $\bar{\mathcal{H}}_i \subseteq \mathbb{N}_d \times \{i\}$. Under the conditions of Assumption 1, the followings hold with probability larger than $1-\eta$:*

$$\|\bar{\epsilon}_{\bar{\mathcal{H}}_i}\|^2 \leq \sigma^2\rho_i^+(|\bar{\mathcal{H}}_i|)(7.4|\bar{\mathcal{H}}_i| + 2.7\ln(2/\eta))/n.$$

Appendix C. MSMTFL-type Algorithms

We present the multi-stage (-type) algorithms for the formulations in Equation (3) and Equation (4) below.

References

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Algorithm 2: MSMTFL-CapL1: Multi-Stage Multi-Task Feature Learning for solving the capped-ℓ_1 regularized feature learning problem in Equation (3)

Initialize $\lambda_j^{(0)} = \lambda$;
for $\ell = 1, 2, \cdots$ **do**
  Let $\hat{W}^{(\ell)}$ be a solution of the following problem:

$$\min_{W \in \mathbb{R}^{d \times m}} \left\{ l(W) + \sum_{j=1}^{d} \lambda_{ji}^{(\ell-1)} |w_{ji}| \right\}.$$

  Let $\lambda_{ji}^{(\ell)} = \lambda I(|\hat{w}_{ji}^{(\ell)}| < \theta)$ $(j = 1, \cdots, d, i = 1, \cdots, m)$, where $\hat{w}_{ji}^{(\ell)}$ is the (j,i)-th entry of $\hat{W}^{(\ell)}$ and $I(\cdot)$ denotes the $\{0,1\}$-valued indicator function.
end

Algorithm 3: MSMTFL-CapL1,L2: Multi-Stage Multi-Task Feature Learning for solving the capped-ℓ_1, ℓ_2 regularized multi-task feature learning problem in Equation (4)

Initialize $\lambda_j^{(0)} = \lambda$;
for $\ell = 1, 2, \cdots$ **do**
  Let $\hat{W}^{(\ell)}$ be a solution of the following problem:

$$\min_{W \in \mathbb{R}^{d \times m}} \left\{ l(W) + \sum_{j=1}^{d} \lambda_j^{(\ell-1)} \|\mathbf{w}^j\| \right\}.$$

  Let $\lambda_j^{(\ell)} = \lambda I(\|(\hat{\mathbf{w}}^{(\ell)})^j\| < \theta)$ $(j = 1, \cdots, d)$, where $(\hat{\mathbf{w}}^{(\ell)})^j$ is the j-th row of $\hat{W}^{(\ell)}$ and $I(\cdot)$ denotes the $\{0,1\}$-valued indicator function.
end

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

J. Bi, T. Xiong, S. Yu, M. Dundar, and R. Rao. An improved multi-task learning approach with applications in medical diagnosis. *Machine Learning and Knowledge Discovery in Databases*, pages 117–132, 2008.

E.J. Candes and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

J. Chen, J. Liu, and J. Ye. Learning incoherent sparse and low-rank patterns from multiple tasks. In *SIGKDD*, pages 1179–1188, 2010.

D.L. Donoho, M. Elad, and V.N. Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory*, 52(1):6–18, 2006.

T. Evgeniou and M. Pontil. Regularized multi–task learning. In *SIGKDD*, pages 109–117, 2004.

J. Fan, L. Xue, and H. Zou. Strong oracle optimality of folded concave penalized estimation. *ArXiv Preprint ArXiv:1210.5992*, 2012.

P. Gong, J. Ye, and C. Zhang. Robust multi-task feature learning. In *SIGKDD*, pages 895–903, 2012.

P. Gong, C. Zhang, Z. Lu, J. Huang, and J. Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *ICML*, 2013a.

P. Gong, C. Zhang, Z. Lu, J. Huang, and J. Ye. *GIST: General Iterative Shrinkage and Thresholding for Non-convex Sparse Learning*. Tsinghua University, 2013b. URL `http://www.public.asu.edu/~jye02/Software/GIST`.

L. Grippo and M. Sciandrone. On the convergence of the block nonlinear gauss-seidel method under convex constraints. *Operations Research Letters*, 26(3):127–136, 2000.

J. Huang and T. Zhang. The benefit of group sparsity. *The Annals of Statistics*, 38(4):1978–2004, 2010.

A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A dirty model for multi-task learning. In *NIPS*, pages 964–972, 2010.

S. Kim and E.P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *ICML*, pages 543–550, 2009.

M. Kolar, J. Lafferty, and L. Wasserman. Union support recovery in multi-task learning. *Journal of Machine Learning Research*, 12:2415–2435, 2011.

J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization. In *UAI*, pages 339–348, 2009.

K. Lounici, M. Pontil, A.B. Tsybakov, and S. Van De Geer. Taking advantage of sparsity in multi-task learning. In *COLT*, pages 73–82, 2009.

S. Negahban and M.J. Wainwright. Joint support recovery under high-dimensional scaling: Benefits and perils of $\ell_{1,\infty}$-regularization. In *NIPS*, pages 1161–1168, 2008.

S. Negahban and M.J. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *The Annals of Statistics*, 39(2):1069–1097, 2011.

G. Obozinski, B. Taskar, and M.I. Jordan. Multi-task feature selection. Technical report, Statistics Department, UC Berkeley, 2006.

G. Obozinski, M.J. Wainwright, and M.I. Jordan. Support union recovery in high-dimensional multivariate regression. *Annals of Statistics*, 39(1):1–47, 2011.

S. Parameswaran and K. Weinberger. Large margin multi-task metric learning. In *NIPS*, pages 1867–1875, 2010.

N. Quadrianto, A. Smola, T. Caetano, SVN Vishwanathan, and J. Petterson. Multitask learning without label correspondences. In *NIPS*, pages 1957–1965, 2010.

R.T. Rockafellar. *Convex Analysis*. Princeton University Press (Princeton, NJ), 1970.

A. Schwaighofer, V. Tresp, and K. Yu. Learning gaussian process kernels via hierarchical bayes. In *NIPS*, pages 1209–1216, 2005.

X. Shen, W. Pan, and Y. Zhu. Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107(497):223–232, 2012.

R. Tibshirani. The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7:1456–1490, 2013.

J.F. Toland. A duality principle for non-convex optimisation and the calculus of variations. *Archive for Rational Mechanics and Analysis*, 71(1):41–61, 1979.

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

S.A. Van De Geer and P. Bühlmann. On the conditions used to prove oracle results for the lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2009.

S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

X. Yang, S. Kim, and E. Xing. Heterogeneous multitask learning with joint sparsity constraints. In *NIPS*, pages 2151–2159, 2009.

K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *ICML*, pages 1012–1019, 2005.

A.L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.

C.H. Zhang and J. Huang. The sparsity and bias of the lasso selection in high-dimensional linear regression. *The Annals of Statistics*, 36(4):1567–1594, 2008.

C.H. Zhang and T. Zhang. A general theory of concave regularization for high dimensional sparse estimation problems. *Statistical Science*, 2012.

J. Zhang, Z. Ghahramani, and Y. Yang. Learning multiple related tasks using latent independent component analysis. In *NIPS*, pages 1585–1592, 2006.

T. Zhang. Adaptive forward-backward greedy algorithm for sparse learning with linear models. In *NIPS*, pages 1921–1928, 2008.

T. Zhang. Some sharp performance bounds for least squares regression with ℓ_1 regularization. *The Annals of Statistics*, 37:2109–2144, 2009.

T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1081–1107, 2010.

T. Zhang. Multi-stage convex relaxation for feature selection. *Bernoulli*, 2012.

Y. Zhang and D.Y. Yeung. Multi-task learning using generalized t process. *Journal of Machine Learning Research - Proceedings Track*, 9:964–971, 2010.

Y. Zhang, D. Yeung, and Q. Xu. Probabilistic multi-task feature selection. In *NIPS*, pages 2559–2567, 2010.

J. Zhou, J. Chen, and J. Ye. Clustered multi-task learning via alternating structure optimization. In *NIPS*, pages 702–710, 2011.

H. Zou and R. Li. One-step sparse estimates in nonconcave penalized likelihood models. *Annals of Statistics*, 36(4):1509, 2008.

Journal of Machine Learning Research 14 (2013) 3011-3040 Submitted 11/12; Revised 9/13; Published 10/13

A Plug-in Approach to Neyman-Pearson Classification

Xin Tong XINT@MARSHALL.USC.EDU
Marshall Business School
University of Southern California
Los Angeles, CA 90089, USA

Editor: John Shawe-Taylor

Abstract

The Neyman-Pearson (NP) paradigm in binary classification treats type I and type II errors with different priorities. It seeks classifiers that minimize type II error, subject to a type I error constraint under a user specified level α. In this paper, plug-in classifiers are developed under the NP paradigm. Based on the fundamental Neyman-Pearson Lemma, we propose two related plug-in classifiers which amount to thresholding respectively the class conditional density ratio and the regression function. These two classifiers handle different sampling schemes. This work focuses on theoretical properties of the proposed classifiers; in particular, we derive oracle inequalities that can be viewed as finite sample versions of risk bounds. NP classification can be used to address anomaly detection problems, where asymmetry in errors is an intrinsic property. As opposed to a common practice in anomaly detection that consists of thresholding normal class density, our approach does not assume a specific form for anomaly distributions. Such consideration is particularly necessary when the anomaly class density is far from uniformly distributed.

Keywords: plug-in approach, Neyman-Pearson paradigm, nonparametric statistics, oracle inequality, anomaly detection

1. Introduction

Classification aims to identify which category a new observation belongs to, on the basis of labeled training data. Applications include disease classification using high-throughput data such as microarrays, SNPs, spam detection and image recognition. This work investigates Neyman-Pearson paradigm in classification with a plug-in approach.

1.1 Neyman-Pearson Paradigm

The Neyman-Pearson (NP) paradigm extends the objective of classical binary classification in that, while the latter focuses on minimizing classification error that is a weighted sum of type I and type II errors, the former minimizes type II error subject to an upper bound α on type I error, where the threshold level α is chosen by the user. The NP paradigm is appropriate in many applications where it is necessary to bring down one kind of error at the expense of the other. One example is medical diagnosis: failing to detect a malignant tumor leads to loss of a life, while flagging a benign one only induces some unnecessary medical cost. As healthy living and longer life expectancy cannot be compensated by any amount of money, it is desirable to control the false negative rate of any medical diagnosis, perhaps with some sacrifice in the false positive rate.

A few commonly used notations in classification literature are set up to facilitate our discussion. Let (X,Y) be a random couple where $X \in \mathcal{X} \subset \mathbb{R}^d$ is a vector of covariates, and where $Y \in \{0,1\}$ is a label that indicates to which class X belongs. A *classifier* h is a mapping $h : \mathcal{X} \to \{0,1\}$ that returns the predicted class given X. An error occurs when $h(X) \neq Y$. It is therefore natural to define the classification loss by $\mathbb{1}(h(X) \neq Y)$, where $\mathbb{1}(\cdot)$ denotes the indicator function. The expectation of the classification loss with respect to the joint distribution of (X,Y) is called *classification risk (error)* and is defined by

$$R(h) = \mathbb{P}\left(h(X) \neq Y\right).$$

The risk function can be expressed as a convex combination of type I and II errors:

$$R(h) = \mathbb{P}(Y=0)R_0(h) + \mathbb{P}(Y=1)R_1(h), \tag{1}$$

where

$$R_0(h) = \mathbb{P}\left(h(X) \neq Y | Y=0\right) \text{denotes the type I error,}$$
$$R_1(h) = \mathbb{P}\left(h(X) \neq Y | Y=1\right) \text{denotes the type II error.}$$

Also recall that the regression function of Y on X is defined by

$$\eta(x) = \mathbb{E}[Y|X=x] = \mathbb{P}(Y=1|X=x).$$

Let $h^*(x) = \mathbb{1}(\eta(x) \geq 1/2)$. The oracle classifier h^* is named the Bayes classifier, and it achieves the minimum risk among all possible candidate classifiers. The risk of h^*, $R^* = R(h^*)$ is called the Bayes risk. A certain classifier $\hat{h}$ in classical binary classification paradigm is good if the excess risk $R(\hat{h}) - R^*$ is small on average or with high probability.

In contrast to the classical paradigm, the NP classification seeks a minimizer ϕ^* that solves

$$\min_{R_0(\phi) \leq \alpha} R_1(\phi),$$

where a small α (e.g., 5%) reflects very conservative attitude towards type I error.

The NP paradigm is irrelevant if we can achieve very small type I and type II errors simultaneously. This is often impossible as expected, and we will demonstrate this point with a stylized example. Note that for most joint distributions on (X,Y), the Bayes error R^* is well above zero. Suppose in a tumor detection application, $R^* = 10\%$. Clearly by (1), it is not feasible to have both type I error R_0 and type II error R_1 be smaller than 10%. Since we insist on lowering the false negative rate as our priority, with a desirable false negative rate much lower than 10%, we have to sacrifice some false positive rate.

Moreover, even if a classifier $\hat{\phi}$ achieves a small risk, there is no guarantee on attaining desirable type I or type II errors. Take another stylized example in medical diagnosis. Suppose that type I error equals 0.5, that is, with 50% of the chances, detector $\hat{\phi}$ fails to find the malignant tumor, and that type II error equals 0.01. Also assume the chance that a tumor is malignant is only 0.001. Then the risk of $\hat{\phi}$ is approximately 1%. This is low, but $\hat{\phi}$ is by no means a good detector, because it misses a malignant tumor with half of the chances!

Empirical risk minimization (ERM), a common approach to classification, has been studied in the NP classification literature. Cannon et al. (2002) initiated the theoretical treatment of the

NP classification paradigm and an early empirical study can be found in Casasent and Chen (2003). Several results for traditional statistical learning such as PAC bounds or oracle inequalities have been studied in Scott (2005) and Scott and Nowak (2005) in the same framework as the one laid down by Cannon et al. (2002). Scott (2007) proposed performance measures for NP classification that weights type I and type II error in sensible ways. More recently, Blanchard et al. (2010) developed a general solution to semi-supervised novelty detection by reducing it NP classification, and Han et al. (2008) transposed several earlier results to NP classification with convex loss. There is a commonality in this line of literature: a relaxed empirical type I error constraint is used in the optimization program, and as a result, type I errors of the classifiers can only be shown to satisfy a relaxed upper bound. Take the framework set up by Cannon et al. (2002) for example: for some $\varepsilon_0 > 0$ and let $\mathcal{H}$ be a set of classifiers with finite VC dimension. They proposed the program

$$\min_{\phi \in \mathcal{H}, \hat{R}_0(\phi) \leq \alpha + \varepsilon_0/2} \hat{R}_1(\phi),$$

where $\hat{R}_0$ and $\hat{R}_1$ denote empirical type I and type II errors respectively. It is shown that solution to the above program $\hat{\phi}$ satisfies simultaneously with high probability, the type II error $R_1(\hat{\phi})$ is bounded from above by $R_1(\phi^*) + \varepsilon_1$, for some $\varepsilon_1 > 0$, and the type I error $R_0(\hat{\phi})$ is bounded from above by $\alpha + \varepsilon_0$.

However, following the original spirit of NP classification, a good classifier $\hat{\phi}$ should respect the chosen significance level α, rather than some relaxation of α, that is, we should be able to i). satisfy the type I error constraint $R_0(\hat{\phi}) \leq \alpha$ with high probability, while ii). establishing an explicit diminishing rate for the excess type II error $R_1(\hat{\phi}) - R_1(\phi^*)$. The simultaneous achievements of i). and ii). can be thought of as counterpart of oracle inequality in classical binary classification, and we believe they are a desirable formulation of theoretical properties of good classifiers in NP classification. Considering this point, Rigollet and Tong (2011) propose a computationally feasible classifier $\tilde{h}^\tau$, such that φ-type I error of $\tilde{h}^\tau$ is bounded from above by α with high probability and the excess φ-type II error of $\tilde{h}^\tau$ converges to 0 with explicit rates, where φ-type I error and φ-type II error are standard convex relaxations of type I and type II errors respectively. Most related to the current context, they also proved a negative result. Loosely speaking, it is shown by counter examples that under the original type I/II criteria, if one adopts ERM approaches (convexification or not), one cannot guarantee diminishing excess type II error if one insists type I error of the proposed classifier be bounded from above by α with high probability. Interested readers are referred to Section 4.4 of that paper.

In this work, we will fulfill the original NP paradigm spirit with the plug-in approach. Theoretical properties of the classifiers under the NP paradigm will be derived. To the best of our knowledge, our paper is the first to do so. It looks as if from a theoretical point of view, a plug-in approach is more suitable than ERM for the NP paradigm. However, such a comparison is not fair because the two approaches are based on different sets of assumptions. For the ERM approach, the main assumption is on the complexity of candidate classifiers, leaving the class conditional distributions unrestricted. While with the plug-in approach, we put restrictions on the joint distributions.

A related framework that also addresses asymmetry in errors is the cost-sensitive learning, which assigns different costs as weights of type I and type II errors (see, e.g., Elkan 2001, Zadrozny et al. 2003). This approach has many practical values, but when it is hard to assign costs to errors, or in applications such as medical diagnosis, where it is morally inappropriate to do the usual cost and benefit analysis, the NP paradigm is a natural choice.

1.2 Plug-in Approach Based on the Fundamental Neyman-Pearson Lemma

NP classification is closely related to the NP approach to statistical hypothesis testing. The punch line is that the fundamental Neyman-Pearson lemma itself suggests a direct plug-in classifier. The interested reader is referred to Lehmann and Romano (2005) for a comprehensive treatment of hypothesis testing. Here we only review the central knowledge that brings up this connection.

Hypothesis testing bears strong resemblance with binary classification if we assume the following model. Let P^- and P^+ be two *known* probability distributions on $\mathcal{X} \subset \mathbb{R}^d$. Let $\pi \in (0,1)$ and assume that Y is a random variable defined by

$$Y = \begin{cases} 1 & \text{with probability } \pi, \\ 0 & \text{with probability } 1-\pi. \end{cases}$$

Assume further that the conditional distribution of X given Y is denoted by P^{2Y-1}. Given such a model, the goal of statistical hypothesis testing is to determine whether X was generated from P^- or from P^+. To that end, we construct a randomized test $\phi : \mathcal{X} \to [0,1]$ and the conclusion of the test based on ϕ is that X is generated from P^+ with probability $\phi(X)$ and from P^- with probability $1-\phi(X)$. Note that randomness here comes from an exogenous randomization process such as flipping a biased coin. Two kinds of errors arise: type I error occurs when rejecting P^- when it is true, and type II error occurs when not rejecting P^- when it is false. The Neyman-Pearson paradigm in hypothesis testing amounts to choosing ϕ that solves the following constrained optimization problem

$$\begin{array}{ll} \text{maximize} & \mathbb{E}[\phi(X)|Y=1], \\ \text{subject to} & \mathbb{E}[\phi(X)|Y=0] \le \alpha, \end{array}$$

where $\alpha \in (0,1)$ is the significance level of the test. In other words, we specify a significance level α on type I error, and minimize type II error. We call a solution to this constrained optimization problem *a most powerful test* of level α. The Neyman-Pearson Lemma gives mild sufficient conditions for the existence of such a test.

Theorem 1 (Neyman-Pearson Lemma) *Let P^- and P^+ be probability distributions possessing densities p_0 and p_1 respectively with respect to some measure μ. Let $f_{C_\alpha}(x) = \mathbb{I}(L(x) \ge C_\alpha)$, where $L(x) = p_1(x)/p_0(x)$ and C_α is such that $P^-(L(X) > C_\alpha) \le \alpha$ and $P^-(L(X) \ge C_\alpha) \ge \alpha$. Then,*

- *f_{C_α} is a level $\alpha = \mathbb{E}[f_{C_\alpha}(X)|Y=0]$ most powerful test.*
- *For a given level α, the most powerful test of level α is defined by*

$$\phi(X) = \begin{cases} 1 & \text{if} \quad L(X) > C_\alpha \\ 0 & \text{if} \quad L(X) < C_\alpha \\ \frac{\alpha - P^-(L(X) > C_\alpha)}{P^-(L(X) = C_\alpha)} & \text{if} \quad L(X) = C_\alpha. \end{cases}$$

Notice that in the learning framework, ϕ cannot be computed since it requires knowledge of the distributions P^- and P^+. Nevertheless, the Neyman-Pearson Lemma motivates a plug-in classifier. Concretely, although we do not know p_1 and p_0, we can find the kernel density estimators $\hat{p}_1$ and $\hat{p}_0$ based on data. Then if we can also detect the approximately right threshold level $\hat{C}_\alpha$, the plug-in approach leads to a classifier $\mathbb{I}(\hat{p}_1(x)/\hat{p}_0(x) \ge \hat{C}_\alpha)$. We expect that this simple classifier would have good type I/II performance bounds, and this intuition will be verified in the following sections. It

is worthy to note that our plug-in approach to NP classification leads to problems related to density level set estimation (see Rigollet and Vert 2009 and reference therein), where the task is to estimate $\{x : p(x) > \lambda\}$, for some level $\lambda > 0$. Density level set estimation has applications in anomaly detection and unsupervised or semi-supervised classification. Plug-in methods for density level set estimation, as opposed to direct methods, do not involve complex optimization procedure, and only amounts to thresholding the density estimate at proper level. The challenges in our setting different from Rigollet and Vert (2009) are two folds. First, the threshold level in our current setup needs to be estimated, and secondly, we deal with density ratios rather than densities. Plug-in methods in classical binary classification have been also studied in the literature. Earlier works seemed to give rise to pessimism of plug-in approach to classification. For example, under certain assumptions, Yang (1999) showed plug-in estimators cannot achieve classification error faster than $O(1/\sqrt{n})$. But direct methods can achieve fast rates up to $O(1/n)$ under *margin assumption* (Mammen and Tsybakov, 1999; Tsybakov, 2004; Tsybakov and van de Geer, 2005; Tarigan and van de Geer, 2006). However Audibert and Tsybakov (2007) combined a smoothness condition condition on regression function with the margin assumption, and showed that plug-in classifiers $\mathbb{I}(\hat{\eta}_n \geq 1/2)$ based on local polynomial estimators can achieve rates faster than $O(1/n)$. We will borrow the smoothness condition on the regression function and margin assumption from Audibert and Tsybakov (2007). However in that paper again, the threshold level is not estimated, so new techniques are called for.

1.3 Application to Anomaly Detection

NP classification is a useful framework to address anomaly detection problems. In anomaly detection, the goal is to discover patterns that are different from usual outcomes or behaviors. An unusual behavior is named an *anomaly*. A variety of problems, such as credit card fraud detection, insider trading detection and system malfunctioning diagnosis, fall into this category. There are many approaches to anomaly detection; some serving a specific purpose while others are more generic. Modeling techniques include classification, clustering, nearest neighbors, statistical and spectrum, etc. A recent comprehensive review of anomaly detection literature is provided by Chandola et al. (2009). Earlier review papers include Agyemang et al. (2006), Hodge and Austin (2004), Markou and Singh (2003a), Markou and Singh (2003b), Patcha and Park (2007), etc.

When we have training data from the normal class, a common approach to anomaly detection is to estimate the normal class density p_0 and try to threshold at a proper level, but this is inappropriate if the anomaly class is far from uniformly distributed. Indeed, to decide whether a certain point is an anomaly, one should consider how likely it is for this point to be normal as opposed to abnormal. The likelihood ratio p_1/p_0 or the regression function η are good to formalize such a concern. Our main results in NP classification will be adapted for anomaly detection applications, where the normal sample size n is much bigger than the anomaly sample size m.

The rest of the paper is organized as follows. In Section 2, we introduce a few notations and definitions. In Section 3, oracle inequalities for a direct plug-in classifier are derived based on the density ratio p_1/p_0. Section 4 investigates another related plug-in classifier, which targets on the regression function η. Finally, proofs of two important technical lemmas are relegated to the Appendix.

2. Notations and Definitions

Following Audibert and Tsybakov (2007), some notations are introduced. For any multi-index $s = (s_1, \ldots, s_d) \in \mathbb{N}^d$ and any $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$, define $|s| = \sum_{i=1}^d s_i$, $s! = s_1! \cdots s_d!$, $x^s = x_1^{s_1} \cdots x_d^{s_d}$ and $\|x\| = (x_1^2 + \cdots + x_d^2)^{1/2}$. Let D^s be the differential operator $D^s = \frac{\partial^{s_1+\cdots+s_d}}{\partial x_1^{s_1} \ldots \partial x_d^{s_d}}$.

Let $\beta > 0$. Denote by $\lfloor \beta \rfloor$ the largest integer strictly less than β. For any $x, x' \in \mathbb{R}^d$ and any $\lfloor \beta \rfloor$ times continuously differentiable real valued function g on $\mathbb{R}^d$, we denote by g_x its Taylor polynomial of degree $\lfloor \beta \rfloor$ at point x:

$$g_x(x') = \sum_{|s| \le \lfloor \beta \rfloor} \frac{(x'-x)^s}{s!} D^s g(x) .$$

For $L > 0$, the $(\beta, L, [-1,1]^d)$-Hölder class of functions, denoted by $\Sigma(\beta, L, [-1,1]^d)$, is the set of functions $g : \mathbb{R}^d \to \mathbb{R}$ that are $\lfloor \beta \rfloor$ times continuously differentiable and satisfy, for any $x, x' \in [-1,1]^d$, the inequality:

$$|g(x') - g_x(x')| \le L\|x - x'\|^\beta .$$

The $(\beta, L, [-1,1]^d)$-Hölder class of density is defined as

$$\mathcal{P}_\Sigma(\beta, L, [-1,1]^d) = \left\{ p : p \ge 0, \int p = 1, p \in \Sigma(\beta, L, [-1,1]^d) \right\} .$$

Denote respectively by $\mathbb{P}$ and $\mathbb{E}$ generic probability distribution and expectation. Also recall that we have denoted by p_0 the density of class 0 and by p_1 that of class 1. For all the theoretical discussions in this paper, the domain of densities p_0 and p_1 is $[-1,1]^d$.

We will use β-valid kernels throughout the paper, which are a multi-dimensional analog of univariate higher order kernels. The definition of β-valid kernels is as follows

Definition 1 *Let K be a real-valued function on $\mathbb{R}^d$ with support $[-1,1]^d$. For fixed $\beta > 0$, the function $K(\cdot)$ is a β-valid kernel if it satisfies $\int K = 1$, $\int |K|^p < \infty$ for any $p \ge 1$, $\int \|t\|^\beta |K(t)| dt < \infty$, and in the case $\lfloor \beta \rfloor \ge 1$, it satisfies $\int t^s K(t) dt = 0$ for any $s = (s_1, \ldots, s_d) \in \mathbb{N}^d$ such that $1 \le s_1 + \ldots + s_d \le \lfloor \beta \rfloor$.*

One example of β-valid kernels is the product kernel whose ingredients are kernels of order β in 1 dimension:

$$\tilde{K}(x) = K(x_1)K(x_2)\cdots K(x_d)\, \mathbb{I}(x \in [-1,1]^d),$$

where K is a 1-dimensional β-valid kernel and is constructed based on Legendre polynomials. We refer interested readers to Section 1.2.2 of Tsybakov (2009). These kernels have been considered in the literature, such as Rigollet and Vert (2009). When the β-valid kernel K is constructed out of Legendre polynomials, it is also Lipschitz and bounded. Therefore, such a kernel satisfies conditions for Lemma 1. For simplicity, we assume that all the β-valid kernels considered in this paper are constructed from Legendre polynomials.

The next low noise condition helps characterize the difficulty of a classification problem.

Definition 2 (Margin Assumption) *A function p satisfies the margin assumption of order $\bar{\gamma}$ with respect to probability distribution P at the level C^* if there exist positive constants C_0 and $\bar{\gamma}$, such that $\forall \delta \ge 0$,*

$$P(|p(X) - C^*| \le \delta) \le C_0 \delta^{\bar{\gamma}}.$$

The above condition for densities was first introduced in Polonik (1995), and its counterpart in the classical binary classification was called margin condition (Mammen and Tsybakov, 1999), from which we borrow the same terminology for discussion. A classification problem is less noisy by requiring most data be further away from the optimal decision boundary. Recall that the set $\{x : \eta(x) = 1/2\}$ is the decision boundary of the Bayes classifier in the classical paradigm, and the margin condition in the classical paradigm is a special case of Definition 2 by taking $p = \eta$ and $C^* = 1/2$.

3. Plug-in Based on Ratio of Class Conditional Densities

In this section, we investigate a plug-in classifier motivated by the Neyman-Pearson Lemma based on the density ratio p_1/p_0. Both the p_0 known and the p_0 unknown cases will be discussed. Although assuming precise knowledge on class 0 density is far from realistic, the subtlety of the plug-in approach in the NP paradigm, as opposed to in the classical paradigm, is revealed through the comparison of the two cases. Most importantly, we formulate some *detection condition* to detect the right threshold level in plug-in classifiers under the NP paradigm.

3.1 Class 0 Density p_0 Known

In this subsection, suppose that we know the class 0 density p_0, but have to estimate the class 1 density p_1. It is interesting to note that this setup is essentially a dual of generalized quantile (minimum volume) set estimation problems, where the volume and mass defining measures are interchanged. Denote by $\hat{p}_1$ the kernel density estimator of p_1 based on an i.i.d. class 1 sample $\mathcal{S}_1 = \{X_1^+, \ldots, X_m^+\}$, that is,

$$\hat{p}_1(x_0) = \frac{1}{mh^d} \sum_{i=1}^{m} K\left(\frac{X_i^+ - x_0}{h}\right),$$

where h is the bandwidth. For a given level α, define respectively $\hat{C}_\alpha$ and C_α^* as solutions of

$$P_0\left(\frac{\hat{p}_1(X)}{p_0(X)} \geq \hat{C}_\alpha\right) = \alpha \qquad \text{and} \qquad P_0\left(\frac{p_1(X)}{p_0(X)} \geq C_\alpha^*\right) = \alpha\,.$$

Note that for some α, $\hat{C}_\alpha$ and C_α^* might not exist. In such cases, randomization is needed to achieve the exact level α. For simplicity, we assume that $\hat{C}_\alpha$ and C_α^* exist and are unique. Note that since p_0 is known, the threshold $\hat{C}_\alpha$ is detected precisely for each sample $\mathcal{S}_1$. The Neyman-Pearson Lemma says that under mild regularity conditions, $\phi^*(x) = \mathbb{I}(p_1(x)/p_0(x) \geq C_\alpha^*)$ is the most powerful test of level α. Therefore, we have a plug-in classifier naturally motivated by the Neyman-Pearson Lemma:

$$\hat{\phi}(x) = \mathbb{I}\left(\frac{\hat{p}_1(x)}{p_0(x)} \geq \hat{C}_\alpha\right), \tag{2}$$

where we plug in estimates $\hat{p}_1$ and $\hat{C}_\alpha$ respectively for the class 1 density p_1 and the threshold level C_α^*. We are interested in the theoretical properties of $\hat{\phi}$. In particular, we will establish oracle inequalities regarding the excess type I and type II errors. Note that since $\hat{C}_\alpha$ is constructed to meet the level α exactly, the excess type I error of $\hat{\phi}$ vanishes, that is,

$$R_0(\hat{\phi}) - R_0(\phi^*) = 0\,.$$

We summarize as follows assumptions on class conditional densities that we will reply upon.

Condition 1 *Suppose that the class conditional densities p_0 and p_1 satisfy:*

i) There exists a positive constant $\mu_{\min}$, such that $p_0 \geq \mu_{\min}$,

ii) The class 1 density $p_1 \in \mathcal{P}_\Sigma(\beta, L, [-1,1]^d)$,

iii) The ratio of class conditional densities p_1/p_0 satisfies the margin assumption of order $\bar{\gamma}$ with respect to probability distribution P_0 at the level C^*_α.

Note that part i) in Condition 1 is the same as assuming $p_0 > 0$ on the compact domain $[-1,1]^d$, as long as p_0 is continuous. Part ii) is a global smoothness condition, which is stronger than the local smoothness conditions used in Rigollet and Vert (2009), in which different smoothness conditions for a neighborhood around the interested level λ and for the complement of the neighborhood are formulated. Rigollet and Vert (2009) emphasized on the smoothness property for a neighborhood around level λ, as only this part affects the rate of convergence. However, as $\hat{C}_\alpha$ is not known a priori in our setup, we rely upon a global smoothness condition as opposed to a local one.

The following theorem addresses the excess type II error of $\hat{\phi}$: $R_1(\hat{\phi}) - R_1(\phi^*)$.

Proposition 1 *Let $\hat{\phi}$ be the plug-in classifier defined by (2). Assume that the class conditional densities p_0 and p_1 satisfy the Condition 1 and that the kernel K is β-valid and L'-Lipschitz. Then for any $\delta \in (0,1)$, and any class 1 sample size m is such that $\sqrt{\frac{\log(m/\delta)}{mh^d}} < 1$, where the bandwidth $h = (\frac{\log m}{m})^{1/(2\beta+d)}$, the excess type II error is bounded, with probability $1-\delta$, by*

$$R_1(\hat{\phi}) - R_1(\phi^*) \leq \frac{2^{2+\bar{\gamma}} C_0 C^{1+\bar{\gamma}}}{(\mu_{\min})^{1+\bar{\gamma}}} \left(\frac{\log(m/\delta)}{mh^d} \right)^{\frac{1+\bar{\gamma}}{2}},$$

where the constant C is the same as in Lemma 1 applied to density p_1. In particular, there exists a positive $\bar{C}$, such that for any $m \geq 1/\delta$,

$$R_1(\hat{\phi}) - R_1(\phi^*) \leq \bar{C} \left(\frac{\log m}{m} \right)^{\frac{\beta(1+\bar{\gamma})}{2\beta+d}}.$$

Note that the dependency of the upper bound for the excess type II error on parameters β, L, and L' is incorporated into the constant C, whose explicit formula is given in Lemma 1, which has an important role in the proof. Lemma 1 is a finite sample uniform deviation result on kernel density estimators. Here we digress slightly and remark that theoretical properties of kernel density estimators have been studied intensively in the literature. A result of similar flavor was obtained in Lei et al. (2013). Readers are referred to Wied and Weiβbach (2010) and references therein for a survey on consistency of kernel density estimators. Convergence in distribution for weighted sup norms was derived in Giné et al. (2004). Lepski (2013) studied expected sup-norm loss of multivariate density estimation with an oracle approach. We have the technical Lemma 1 and it proof in the appendix, as none of previous results is tailored to our use. Another phenomenon worth mentioning is that the upper bound does not explicitly depend on the significance level α. This results from the way we formulate the margin assumption. Suppose we were to allow $\bar{\gamma}$ in the margin assumption to depend on α, that is, $\bar{\gamma} = \bar{\gamma}(\alpha)$, or let C_0 depend on α, the upper bound would have explicit dependency on α. Also from the upper bound, we can see that the larger the parameter $\bar{\gamma}$, the sharper the margin assumption, and then the faster the rate of convergence for the

excess type II error. Also, we re-emphasize that the feature dimension d considered in this paper is fixed and does not increase with sample sizes.

Proof

First note that the excess type II error can be represented by

$$R_1(\hat{\phi}) - R_1(\phi^*) = \int_{G^* \triangle \hat{G}} \left| \frac{p_1}{p_0} - C^*_\alpha \right| dP_0 \,,$$

where $G^* = \left\{ \frac{p_1}{p_0} < C^*_\alpha \right\}$ and $\hat{G} = \left\{ \frac{\hat{p}_1}{p_0} < \hat{C}_\alpha \right\}$, and $G^* \triangle \hat{G} = (G^* \cap \hat{G}^c) \cup (G^{*c} \cap \hat{G})$ is the symmetric difference between G^* and $\hat{G}$. Indeed,

$$\begin{aligned}
& \int_{G^* \triangle \hat{G}} \left| \frac{p_1}{p_0} - C^*_\alpha \right| dP_0 \\
= & \int_{G^* \cap \hat{G}^c} \left(C^*_\alpha - \frac{p_1}{p_0} \right) dP_0 + \int_{G^{*c} \cap \hat{G}} \left(\frac{p_1}{p_0} - C^*_\alpha \right) dP_0 \\
= & \int_{G^*} \left(C^*_\alpha - \frac{p_1}{p_0} \right) dP_0 + \int_{\hat{G}} \left(\frac{p_1}{p_0} - C^*_\alpha \right) dP_0 \\
= & C^*_\alpha P_0(G^*) - P_1(G^*) - C^*_\alpha P_0(\hat{G}) + P_1(\hat{G}) \\
= & P_1(\hat{G}) - P_1(G^*) \,.
\end{aligned}$$

Define an event regarding the sample $\mathcal{S}_1$: $\mathcal{E} = \{ \|\hat{p}_1 - p_1\|_\infty < \frac{\delta_1}{2} \mu_{\min} \}$, where $\delta_1 = \frac{2C}{\mu_{\min}} \sqrt{\frac{\log(m/\delta)}{mh^d}}$, and C is the same as in Lemma 1 (with p replaced by p_1). From this point to the end of the proof, we restrict ourselves to the event $\mathcal{E}$.

Since $G^{*c} \cap \hat{G}$ and $G^* \cap \hat{G}^c$ are disjoint, we can handle the two parts separately. Decompose

$$G^{*c} \cap \hat{G} = \left\{ \frac{p_1}{p_0} \geq C^*_\alpha, \frac{\hat{p}_1}{p_0} < \hat{C}_\alpha \right\} = A_1 \cup A_2 \,,$$

where

$$A_1 = \left\{ C^*_\alpha + \delta_1 \geq \frac{p_1}{p_0} \geq C^*_\alpha, \frac{\hat{p}_1}{p_0} < \hat{C}_\alpha \right\} ,$$

and

$$A_2 = \left\{ \frac{p_1}{p_0} > C^*_\alpha + \delta_1, \frac{\hat{p}_1}{p_0} < \hat{C}_\alpha \right\} .$$

Then,

$$\int_{A_1} \left(\frac{p_1}{p_0} - C^*_\alpha \right) dP_0 \leq \delta_1 P_0(A_1) \leq C_0 (\delta_1)^{1+\bar{\gamma}} .$$

We can control the distance between $\hat{C}_\alpha$ and C^*_α. Indeed,

$$\begin{aligned}
\alpha &= P_0 \left(\frac{\hat{p}_1(X)}{p_0(X)} \geq \hat{C}_\alpha \right) = P_0 \left(\frac{p_1(X)}{p_0(X)} \geq C^*_\alpha \right) \geq P_0 \left(\frac{\hat{p}_1(X)}{p_0(X)} \geq C^*_\alpha + \frac{|p_1(X) - \hat{p}_1(X)|}{p_0(X)} \right) \\
&\geq P_0 \left(\frac{\hat{p}_1(X)}{p_0(X)} \geq C^*_\alpha + \frac{\delta_1 \mu_{\min}}{2\mu_{\min}} \right) = P_0 \left(\frac{\hat{p}_1(X)}{p_0(X)} \geq C^*_\alpha + \frac{\delta_1}{2} \right) .
\end{aligned}$$

This implies that $\hat{C}_\alpha < C^*_\alpha + \frac{\delta_1}{2}$. Therefore,

$$A_2 \subset A_3 := \left\{ \frac{p_1}{p_0} \geq C^*_\alpha + \delta_1, \frac{\hat{p}_1}{p_0} < C^*_\alpha + \delta_1/2 \right\}.$$

It is clear that on the event $\mathcal{E}$, $P_0(A_3) = 0$. Therefore,

$$\int_{G^{*c} \cap \hat{G}} \left(\frac{p_1}{p_0} - C^*_\alpha \right) dP_0 \leq C_0 (\delta_1)^{1+\bar{\gamma}}.$$

Similarly, it can be shown that $\int_{G^* \cap \hat{G}^c} \left(C^*_\alpha - \frac{p_1}{p_0} \right) dP_0 \leq C_0 (\delta_1)^{1+\bar{\gamma}}$. Therefore,

$$\int_{G^* \triangle \hat{G}} \left| \frac{p_1}{p_0} - C^*_\alpha \right| dP_0 \leq 2C_0 (\delta_1)^{1+\bar{\gamma}}.$$

Finally, Lemma 1 implies $\mathbb{P}(\mathcal{E}) \geq 1 - \delta$. This completes the proof.

■

Although it is reasonable to assume that the class 0 density p_0 can be approximated well, assuming p_0 known exactly is not realistic for most applications. Next, we consider the unknown p_0 case.

3.2 Class 0 Density p_0 Unknown

Assume that both the class 0 density p_0 and the class 1 density p_1 are unknown. Knowledge on class conditional densities is passed to us through samples. Because data from class 0 is needed to estimate both the class 0 density and the threshold level, we split the class 0 data into two pieces. Therefore, suppose available data include class 0 samples $\mathcal{S}_0 = \left\{ X_1^-, \ldots, X_n^- \right\}$, $\tilde{\mathcal{S}}_0 = \left\{ X_{n+1}^-, \ldots, X_{2n}^- \right\}$, and a class 1 sample $\mathcal{S}_1 = \left\{ X_1^+, \ldots, X_m^+ \right\}$. Also assume that given samples $\mathcal{S}_0$ and $\mathcal{S}_1$, the variables in $\tilde{\mathcal{S}}_0$ are independent. Our mission is still to construct a plug-in classifier based on the optimal test output by the Neyman-Pearson Lemma and to show that it has desirable theoretical properties regarding type I and II errors.

First estimate p_0 and p_1 respectively from $\mathcal{S}_0$ and $\mathcal{S}_1$ by kernel estimators,

$$\hat{p}_0(x) = \frac{1}{n h_n^d} \sum_{i=1}^{n} K \left(\frac{X_i^- - x}{h_n} \right) \qquad \text{and} \qquad \hat{p}_1(x) = \frac{1}{m h_m^d} \sum_{i=1}^{m} K \left(\frac{X_i^+ - x}{h_m} \right),$$

where h_n and h_m denote the bandwidths. Since p_0 is unknown, $\hat{C}_\alpha$ can not be defined trivially as in the p_0 known case. And, it turns out that detecting the right threshold level is important to proving theoretical properties of the plug-in classifier. There is one essential piece of intuition. We know that having fast diminishing excess type II error demands a low noise condition, such as the margin assumption. On the other hand, if there are enough sample points around the optimal threshold level, we can approximate the threshold C^*_α accurately. Approximating the optimal threshold level is not a problem in the classical setting, because in that setting, the Bayes classifier is $\mathbb{I}(\eta(x) \geq 1/2)$, and the threshold level $1/2$ on the regression function η is known. Therefore, estimating the optimal threshold with the NP paradigm introduces new technical challenges. The following level α detection condition addresses this concern.

Condition 2 (level α detection condition) *The function f satisfies the level α detection condition (with respect to P_0 ($X \sim P_0$)) if there exist positive constants C_1 and γ, such that for any δ in a small right neighborhood of 0,*

$$P_0\left(C_\alpha^* - \delta \le f(X) \le C_\alpha^*\right) \wedge P_0\left(C_\alpha^* \le f(X) \le C_\alpha^* + \delta\right) \ge C_1 \delta^\gamma.$$

Definition 3 *Fix $\delta \in (0,1)$, for $d_n = 2\sqrt{2\frac{\log(2en)+\log(2/\delta)}{n}}$, let $\hat{C}_\alpha$ be the smallest C such that*

$$\frac{1}{n}\sum_{i=n+1}^{2n} \mathbb{I}\left(\frac{\hat{p}_1(X_i^-)}{\hat{p}_0(X_i^-)} \ge C\right) \le \alpha - d_n.$$

Having $\hat{p}_1, \hat{p}_0$ and $\hat{C}_\alpha$, we propose a plug-in classifier motivated by the Neyman-Pearson Lemma in statistical hypothesis testing:

$$\hat{\phi}(x) = \mathbb{I}\left(\frac{\hat{p}_1(x)}{\hat{p}_0(x)} \ge \hat{C}_\alpha\right). \tag{3}$$

Unlike the previous setup where p_0 was known, we now need to bound the type I error of $\hat{\phi}$ first.

Proposition 2 *With probability at least $1-\delta$ regarding the samples $\mathcal{S}_0$, $\tilde{\mathcal{S}}_0$ and $\mathcal{S}_1$, type I error of the plug-in classifier $\hat{\phi}$ defined in* (3) *is bounded from above by α, that is,*

$$R_0(\hat{\phi}) \le \alpha.$$

Proof Note that $R_0(\hat{\phi}) = P_0\left(\frac{\hat{p}_1(X)}{\hat{p}_0(X)} \ge \hat{C}_\alpha\right)$ and $\frac{1}{n}\sum_{i=n+1}^{2n} \mathbb{I}\left(\frac{\hat{p}_1(X_i^-)}{\hat{p}_0(X_i^-)} \ge \hat{C}_\alpha\right) \le \alpha - d_n$. Let

$$\mathcal{A}_t = \left\{ \sup_{c\in\mathbb{R}} \left| P_0\left(\frac{\hat{p}_1(X)}{\hat{p}_0(X)} \ge c\right) - \frac{1}{n}\sum_{i=n+1}^{2n} \mathbb{I}\left(\frac{\hat{p}_1(X_i^-)}{\hat{p}_0(X_i^-)} \ge c\right) \right| \ge t \right\}.$$

Then it is enough to show that, $\mathbb{P}(\mathcal{A}_{d_n}) \le \delta$.

Note that $\mathbb{P}(\mathcal{A}_t) = \mathbb{E}(\mathbb{P}(\mathcal{A}_t | \mathcal{S}_0, \mathcal{S}_1))$. Keep $\mathcal{S}_0$ and $\mathcal{S}_1$ fixed, and define $\hat{f}(x) = \frac{\hat{p}_1(x)}{\hat{p}_0(x)}$. Let Q be the conditional distribution of $Z = \hat{f}(X)$ given $\mathcal{S}_0$ and $\mathcal{S}_1$, where $X \sim P_0$, and Q^n denote the conditional joint distribution of $(Z_{n+1}^-, \ldots, Z_{2n}^-) = (\hat{f}(X_{n+1}^-), \ldots, \hat{f}(X_{2n}^-))$. Because half lines in $\mathbb{R}$ have VC dimension 1, by taking $t = d_n = 2\sqrt{2\frac{\log(2en)+\log(2/\delta)}{n}}$, the VC inequality[1] implies that

$$\mathbb{P}(\mathcal{A}_{d_n} | \mathcal{S}_0, \mathcal{S}_1) \quad = Q^n\left(\sup_c \left| Q(Z \ge c) - \tfrac{1}{n}\textstyle\sum_{i=n+1}^{2n} \mathbb{I}(Z_i^- \ge c) \right| \ge d_n\right) \quad \le \delta.$$

Therefore,

$$\mathbb{P}(\mathcal{A}_{d_n}) = \mathbb{E}(\mathbb{P}(\mathcal{A}_{d_n} | \mathcal{S}_0, \mathcal{S}_1)) \le \delta.$$

■

The next theorem addresses the excess type II error of $\hat{\phi}$.

1. For the readers' convenience, a simple corollary of VC inequality is quoted: let $\mathcal{G}$ be a class of classifiers with VC dimension l, then with probability at least $1-\delta$, $\sup_{g\in\mathcal{G}} |R(g) - R_n(g)| \le 2\sqrt{2\frac{l\log(2en/l)+\log(2/\delta)}{n}}$, where n is the sample size and R_n denotes the empirical risk.

Theorem 2 *Let $\hat{\phi}$ be the plug-in classifier defined as in* (3)*. Assume that the class conditional densities p_0 and p_1 satisfy Condition 1, $p_0 \in \mathcal{P}_\Sigma(\beta, L, [-1,1]^d)$ and the kernel K is β-valid and L'-Lipschitz. Also assume that the likelihood ratio p_1/p_0 satisfies the level α detection condition for some $\underline{\gamma} \geq \bar{\gamma}$. Then for any $\delta \in (0,1)$ and any sample sizes m,n such that*

$$\max\left(\sqrt{\frac{\log(m/\delta)}{mh_m^d}}, \sqrt{\frac{\log(n/\delta)}{nh_n^d}}\right) < 1\,,$$

where the bandwidths $h_n = (\frac{\log n}{n})^{\frac{1}{2\beta+d}}$ and $h_m = (\frac{\log m}{m})^{\frac{1}{2\beta+d}}$, it holds with probability $1-3\delta$,

$$R_1(\hat{\phi}) - R_1(\phi^*) \leq 2C_0\left[\left(2d_n/C_1\right)^{1/\underline{\gamma}} + 2T_{m,n}\right]^{1+\bar{\gamma}} + 2C_\alpha^* d_n\,,$$

where $d_n = 2\sqrt{2\frac{\log(2en)+\log(2/\delta)}{n}}$, $T_{m,n} = \frac{\delta_1 + \|p_1\|_\infty \delta_0/\mu_{\min}}{\mu_{\min}-\delta_0}$, $\delta_0 = C_2\sqrt{\frac{\log(n/\delta)}{nh_n^d}}$,
$\delta_1 = C_3\sqrt{\frac{\log(m/\delta)}{mh_m^d}}$, C_2 and C_3 are the same as C in Lemma 1 applied to p_0 and p_1 respectively.

In particular, there exists some positive $\bar{C}$, such that for all $n,m \geq 1/\delta$,

$$R_1(\hat{\phi}) - R_1(\phi^*) \leq \bar{C}\left[\left(\frac{\log n}{n}\right)^{\min\left(\frac{1}{2}, \frac{1+\bar{\gamma}}{2\underline{\gamma}}, \frac{\beta(1+\bar{\gamma})}{2\beta+d}\right)} + \left(\frac{\log m}{m}\right)^{\frac{\beta(1+\bar{\gamma})}{2\beta+d}}\right]. \tag{4}$$

Proof

Denote by $G^* = \left\{\frac{p_1}{p_0} < C_\alpha^*\right\}$ and $\hat{G} = \left\{\frac{\hat{p}_1}{\hat{p}_0} < \hat{C}_\alpha\right\}$. Then

$$\begin{aligned}
&\int_{G^* \triangle \hat{G}} \left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0 \\
= &\int_{G^* \cap \hat{G}^c} \left(C_\alpha^* - \frac{p_1}{p_0}\right) dP_0 + \int_{G^{*c} \cap \hat{G}} \left(\frac{p_1}{p_0} - C_\alpha^*\right) dP_0 \\
= &\int_{G^*} \left(C_\alpha^* - \frac{p_1}{p_0}\right) dP_0 + \int_{\hat{G}} \left(\frac{p_1}{p_0} - C_\alpha^*\right) dP_0 \\
= &P_1(\hat{G}) - P_1(G^*) + C_\alpha^*\left[P_0(G^*) - P_0(\hat{G})\right].
\end{aligned}$$

Therefore the excess type II error can be decomposed in two parts,

$$P_1(\hat{G}) - P_1(G^*) = \int_{G^* \triangle \hat{G}} \left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0 + C_\alpha^*\left[P_0(G^{*c}) - P_0(\hat{G}^c)\right]. \tag{5}$$

Recall that $P_0(G^{*c}) = \alpha$ and $P_0(\hat{G}^c)$ is type I error of $\hat{\phi}$. From the above decomposition, we see that to control the excess type II error, type I error of $\hat{\phi}$ should be not only smaller than the level α, but also not far from α. This is intuitively correct, because having a small type I error amounts to having a very tight constraint set, which leads to significant deterioration in achievable type II error. Fortunately, this is not the case here with high probability. Note that by the definition of $\hat{C}_\alpha$, for any positive number l, the following holds for all $n \geq 1$,

$$\frac{1}{n}\sum_{i=n+1}^{2n} \mathbb{I}\left(\frac{\hat{p}_1(X_i^-)}{\hat{p}_0(X_i^-)} \geq \hat{C}_\alpha - l\right) > \alpha - d_n\,.$$

By the same argument as in the proof of Proposition 2, there exists an event $\bar{\mathcal{E}}_l$ regarding the samples $\mathcal{S}_0$, $\tilde{\mathcal{S}}_0$ and $\mathcal{S}_1$ with $\mathbb{P}(\bar{\mathcal{E}}_l) \geq 1-\delta$, such that on this event,

$$P_0\left(\frac{\hat{p}_1(X)}{\hat{p}_0(X)} \geq \hat{C}_\alpha - l\right) \geq \alpha - 2d_n\,. \tag{6}$$

To control the second part of R.H.S. in(5), let $\hat{G}_l = \{\frac{\hat{p}_1}{\hat{p}_0} < \hat{C}_\alpha - l\}$, then

$$P_0(G^{*c}) - P_0(\hat{G}^c) = \inf_{l>0}\left[P_0(G^{*c}) - P_0(\hat{G}_l^c)\right] \leq \alpha - (\alpha - 2d_n) = 2d_n\,, \tag{7}$$

on the event $\bar{\mathcal{E}} := \cap_{l>0}\bar{\mathcal{E}}_l$, and $\mathbb{P}(\bar{\mathcal{E}}) = \lim_{l\to 0}\mathbb{P}(\bar{\mathcal{E}}_l) \geq 1-\delta$.

Therefore, it remains to control the first part of R.H.S. in (5). Define an event regarding samples $\mathcal{S}_0$ and $\mathcal{S}_1$:

$$\mathcal{E} = \{\|\hat{p}_0 - p_0\|_\infty < \delta_0, \|\hat{p}_1 - p_1\|_\infty < \delta_1\}\,.$$

Lemma 1 implies that $\mathbb{P}(\mathcal{E}) \geq 1-2\delta$. We restrict ourselves to $\mathcal{E}\cap\bar{\mathcal{E}}$ for the rest of the proof. Note that the first part of R.H.S. in (5) can be decomposed by

$$\int_{G^*\triangle\hat{G}}\left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0 \quad = \quad \int_{G^{*c}\cap\hat{G}}\left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0 + \int_{G^*\cap\hat{G}^c}\left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0\,.$$

We will focus on bounding the integral over $G^*\cap\hat{G}^c$, because that over $G^{*c}\cap\hat{G}$ can be bounded similarly. Note that,

$$\left|\frac{\hat{p}_1}{\hat{p}_0} - \frac{p_1}{p_0}\right| \leq \left|\frac{\hat{p}_1}{\hat{p}_0} - \frac{p_1}{\hat{p}_0}\right| + \left|\frac{p_1}{\hat{p}_0} - \frac{p_1}{p_0}\right| \leq \frac{1}{|\hat{p}_0|}|\hat{p}_1 - p_1| + \left|\frac{p_1}{p_0}\right|\cdot\frac{|p_0-\hat{p}_0|}{|\hat{p}_0|} < \frac{\|p_1\|_\infty\delta_0/\mu_{\min} + \delta_1}{\mu_{\min} - \delta_0} = T_{m,n}\,.$$

The above inequality together with (6) implies that

$$\alpha - 2d_n \leq P_0\left(\frac{\hat{p}_1(X)}{\hat{p}_0(X)} \geq \hat{C}_\alpha - l\right) \leq P_0\left(\frac{p_1(X)}{p_0(X)} \geq \hat{C}_\alpha - l - T_{m,n}\right)\,. \tag{8}$$

We need to bound $\hat{C}_\alpha$ in terms of C_α^*. This is achieved through the following steps. First, we determine some $c_n > 0$ such that

$$P_0\left(\frac{p_1(X)}{p_0(X)} \geq C_\alpha^* + c_n\right) \leq \alpha - 2d_n\,,$$

which follows if the next inequality holds

$$2d_n \leq P_0\left(C_\alpha^* < \frac{p_1(X)}{p_0(X)} < C_\alpha^* + c_n\right)\,.$$

By the level α detection condition, it is enough to take $c_n = (2d_n/C_1)^{1/\gamma}$. Therefore in view of inequality (8),

$$P_0\left(\frac{p_1(X)}{p_0(X)} \geq C_\alpha^* + (2d_n/C_1)^{1/\gamma}\right) \leq \alpha - 2d_n \leq P_0\left(\frac{p_1(X)}{p_0(X)} \geq \hat{C}_\alpha - l - T_{m,n}\right)\,.$$

This implies that

$$\hat{C}_\alpha \le C^*_\alpha + (2d_n/C_1)^{1/\gamma} + l + T_{m,n}\,.$$

Since the above holds for all $l > 0$, we have

$$\hat{C}_\alpha \le C^*_\alpha + (2d_n/C_1)^{1/\gamma} + T_{m,n}\,.$$

For any positive $L_{m,n}$, we can decompose $G^{*c} \cap \hat{G}$ by

$$G^{*c} \cap \hat{G} = \left\{ \frac{p_1}{p_0} \ge C^*_\alpha, \frac{\hat{p}_1}{\hat{p}_0} < \hat{C}_\alpha \right\} = A_1 \cap A_2\,,$$

where

$$A_1 = \left\{ C^*_\alpha + L_{m,n} > \frac{p_1}{p_0} \ge C^*_\alpha, \frac{\hat{p}_1}{\hat{p}_0} < \hat{C}_\alpha \right\}, \text{ and } A_2 = \left\{ \frac{p_1}{p_0} \ge C^*_\alpha + L_{m,n}, \frac{\hat{p}_1}{\hat{p}_0} < \hat{C}_\alpha \right\}.$$

By the margin assumption,

$$\int_{A_1} \left| \frac{p_1}{p_0} - C^*_\alpha \right| dP_0 \le L_{m,n} P_0(A_1) \le C_0 (L_{m,n})^{1+\bar{\gamma}}.$$

Note that

$$A_2 \subset A_3 := \left\{ \frac{p_1}{p_0} \ge C^*_\alpha + L_{m,n}, \frac{\hat{p}_1}{\hat{p}_0} < C^*_\alpha + (2d_n/C_1)^{1/\gamma} + T_{m,n} \right\}.$$

Take $L_{m,n} = (2d_n/C_1)^{1/\gamma} + 2T_{m,n}$, then $P_0(A_2) = P_0(A_3) = 0$. Therefore,

$$\int_{G^{*c} \cap \hat{G}} \left| \frac{p_1}{p_0} - C^*_\alpha \right| dP_0 \le C_0 (L_{m,n})^{1+\bar{\gamma}}.$$

Similarly, we can bound the integral over $G^* \cap \hat{G}^c$, so

$$\int_{G^* \triangle \hat{G}} \left| \frac{p_1}{p_0} - C^*_\alpha \right| dP_0 \le 2C_0 (L_{m,n})^{1+\bar{\gamma}}. \tag{9}$$

Finally, note that $\mathbb{P}(\mathcal{E} \cap \bar{\mathcal{E}}) \ge 1 - 3\delta$. So (5), (7) and (9) together conclude the proof.

■

Now we briefly discuss the above result. Same as the p_0 known setup, the coefficient $\bar{\gamma}$ from the margin assumption has influence on the convergence rate of the excess type II error. The larger the $\bar{\gamma}$, the easier the classification problem, and hence the faster the convergence of the excess type II error. The coefficient $\underline{\gamma}$ in the detection condition works differently. The larger the $\underline{\gamma}$, the more difficult it is to detect the optimal decision boundary, and hence the harder the classification problem. Take it to the extreme $\underline{\gamma} \to \infty$ (keep $\bar{\gamma}$ fixed), which holds when the amount of data around the optimal threshold level goes to zero,

$$\left(\frac{\log n}{n} \right)^{\frac{1+\bar{\gamma}}{2\underline{\gamma}}} \to \left(\frac{\log n}{n} \right)^0 = 1\,.$$

In other words, the upper bound in (4) is uninformative when we have a null level α detection condition.

In anomaly detection applications, let class 0 represent the normal class, and class 1 represent the anomaly class. We have in mind $n \gg m$, that is, the normal sample size is much bigger than that of the anomaly, and so $\log n/n$ is dominated by $\log m/m$. Therefore, the right hand side of (4) is of the order $\left[(\frac{\log n}{n})^{\min(1/2,(1+\bar{\gamma})/(2\underline{\gamma}))} + (\frac{\log m}{m})^{\beta(1+\bar{\gamma})/(2\beta+d)} \right]$. Compared with the p_0 known setup, the extra term $\left(\frac{\log n}{n}\right)^{\min(1/2,(1+\bar{\gamma})/(2\underline{\gamma}))}$ arises from estimating the threshold level C^*_α. Let $n \to \infty$, which amounts to knowing p_0, this term vanishes, and the upper bound reduces to the same as in the previous subsection. When $\underline{\gamma} < 1+\bar{\gamma}$, we have $1/2 < (1+\bar{\gamma})/(2\underline{\gamma})$, so $\underline{\gamma}$ does not show up in the upper bound. Finally, for fixed $\underline{\gamma}(\geq 1+\bar{\gamma})$, β and d, we can calculate explicitly an order relation between m and n, such that $(\frac{\log m}{m})^{\frac{\beta}{2\beta+d}} \geq (\frac{\log n}{n})^{\frac{1}{2\underline{\gamma}}}$. A sufficient condition for this inequality is that $n \geq \left(\frac{m}{\log m}\right)^{\beta\underline{\gamma}/(4\beta+2d)}$. Intuitively, this says that if the normal class sample size is large enough compared to the anomaly class sample size, lack of precise knowledge on normal class density p_0 does not change the type II error rate bound, up to a multiplicative constant.

4. Plug-in Based on the Regression Function

In this section, instead of plugging in class conditional densities p_1 and p_0, we target the regression function $\eta(x) = \mathbb{E}(Y|X=x)$ directly. As will be illustrated, this version of plug-in estimator allows us to handle a different assumption on sampling scheme. Recall that the rationality behind plugging in for p_1/p_0 lies in the Neyman-Pearson Lemma for hypothesis testing. A simple derivation shows that a thresholding rule on p_1/p_0 can be translated into a thresholding rule on η. Indeed, let $\pi = \mathbb{P}(Y=1)$, then we have

$$\eta(x) = \mathbb{P}(Y=1|X=x) = \frac{\pi \cdot p_1/p_0(x)}{\pi \cdot p_1/p_0(x) + (1-\pi)}.$$

When $\pi < 1$, the function $x \mapsto \frac{\pi x}{\pi x + (1-\pi)}$ is strictly monotone increasing on $\mathbb{R}^+$. Therefore, there exists a positive constant D^*_α depending on α, such that

$$\left\{ x \in [-1,1]^d : \frac{p_1(x)}{p_0(x)} \geq C^*_\alpha \right\} = \{ x \in [-1,1]^d : \eta(x) \geq D^*_\alpha \}.$$

Moreover, the oracle thresholds C^*_α and D^*_α are related by $D^*_\alpha = \frac{\pi C^*_\alpha}{\pi C^*_\alpha + (1-\pi)}$. Parallel to the previous section, we address both the p_0 known and p_0 unknown setups. In both setups, we assume that we have access to an i.i.d. sample $\bar{\mathcal{S}} = \{(X_1,Y_1),\ldots,(X_m,Y_m)\}$.

4.1 Class 0 Density p_0 Known

This part is similar to the p_0 known setup in Section 3. The essential technical difference is that we need a uniform deviation result on the Nadaraya-Watson estimator $\hat{\eta}$ (based on the sample $\bar{\mathcal{S}}$) instead of those on $\hat{p}_0$ and $\hat{p}_1$. Recall that $\hat{\eta} = \sum_{i=1}^m Y_i K\left(\frac{X_i - x}{h}\right) / \sum_{i=1}^m K\left(\frac{X_i - x}{h}\right)$ can written as $\hat{f}/\hat{p}$,

where

$$\hat{p}(x) = \frac{1}{mh^d}\sum_{i=1}^{m} K\left(\frac{X_i - x}{h}\right) \text{ and } \hat{f}(x) = \frac{1}{mh^d}\sum_{i=1}^{m} Y_i K\left(\frac{X_i - x}{h}\right),$$

in which h is the bandwidth. Denote by $p = \pi p_1 + (1-\pi)p_0$ and $f = \eta \cdot p$, then $\eta = f/p$. For a given level α, define $\hat{D}_\alpha$ and D^*_α respectively by

$$P_0(\hat{\eta}(X) \geq \hat{D}_\alpha) = \alpha \quad \text{and} \quad P_0(\eta(X) \geq D^*_\alpha) = \alpha\,.$$

For simplicity, we assume that $\hat{D}_\alpha$ and D^*_α exist and are unique. Note that the oracle classifier of level α is $\phi^*(x) = \mathbb{I}(\eta(x) \geq D^*_\alpha)$, so a plug-in classifier motivated by the Neyman-Pearson Lemma is

$$\tilde{\phi}(x) = \mathbb{I}(\hat{\eta}(x) \geq \hat{D}_\alpha)\,. \tag{10}$$

Since $\hat{D}_\alpha$ is constructed to meet the level α exactly, the excess type I error of $\tilde{\phi}$ vanishes, that is,

$$R_0(\tilde{\phi}) - R_0(\phi^*) = 0\,.$$

The following theorem addresses type II error of $\tilde{\phi}$.

Condition 3 *Suppose that p the marginal density of X and η the regression function satisfy:*

(i) There exist positive constants $\mu'_{\min}$ and $\nu'_{\max}(<1)$, such that $p \geq \mu'_{\min}$ and $\eta \leq \nu'_{\max}$,

(ii) $f = \eta \cdot p \in \mathcal{P}_\Sigma(\beta, L, [-1,1]^d)$ and $p \in \mathcal{P}_\Sigma(\beta, L, [-1,1]^d)$,

(iii) The regression function η satisfies the margin assumption of order $\bar{\gamma}$ with respect to probability distribution P_0 at the level D^*_α.

Proposition 3 *Let $\tilde{\phi}$ be the plug-in classifier defined by* (10)*. Assume that p and η satisfy condition 3 and that the kernel K is β-valid and L'-Lipschitz. Then there exists a positive $\tilde{D}$, such that for any $\delta \in (0,1)$ and any sample size m satisfying $\sqrt{\frac{\log(m/\delta)}{mh^d}} < 1$, it holds with probability $1-\delta$,*

$$R_1(\tilde{\phi}) - R_1(\phi^*) \leq \tilde{D}\left(\frac{\log(3m/\delta)}{mh^d}\right)^{\frac{1+\bar{\gamma}}{2}},$$

where $h = \left(\frac{\log m}{m}\right)^{\frac{1}{2\beta+d}}$. Furthermore, there exists a positive D such that for any $m \geq 1/\delta$, it holds with probability $1-\delta$,

$$R_1(\tilde{\phi}) - R_1(\phi^*) \leq D\left(\frac{\log m}{m}\right)^{\frac{\beta(1+\bar{\gamma})}{2\beta+d}}.$$

Proof First note that the excess type II error

$$R_1(\tilde{\phi}) - R_1(\phi^*) = \int_{G^* \triangle \hat{G}} \left|\frac{p_1}{p_0} - C^*_\alpha\right| dP_0 = P_1(\hat{G}) - P_1(G^*)\,,$$

where $G^* = \{\eta < D_\alpha^*\}$ and $\hat{G} = \left\{\hat{\eta} < \hat{D}_\alpha\right\}$, and $G^* \triangle \hat{G} = (G^* \cap \hat{G}^c) \cup (G^{*c} \cap \hat{G})$.

Define an event regarding the sample $\bar{S}$,

$$\mathcal{E} = \{\|\hat{\eta} - \eta\|_\infty < \delta_1/2\},$$

where $\delta_1 = D_1\sqrt{\frac{\log(3m/\delta)}{mh^d}}$ and D_1 is a constant chosen as in Lemma 2. From this point to the end of the proof, we restrict ourselves to $\mathcal{E}$. Decompose

$$G^{*c} \cap \hat{G} = \left\{\eta \geq D_\alpha^*, \hat{\eta} < \hat{D}_\alpha\right\} = A_1 \cup A_2,$$

where

$$A_1 = \left\{D_\alpha^* + \delta_1 \geq \eta \geq D_\alpha^*, \hat{\eta} < \hat{D}_\alpha\right\},$$

and

$$A_2 = \left\{\eta > D_\alpha^* + \delta_1, \hat{\eta} < \hat{D}_\alpha\right\}.$$

Now we need to control the distance of $\left|\frac{p_1}{p_0} - C_\alpha^*\right|$ in terms of $|\eta - D_\alpha^*|$. This can be achieved by recalling

$$\eta = \frac{\pi p_1/p_0}{\pi p_1/p_0 + 1 - \pi} \text{ and } D_\alpha^* = \frac{\pi C_\alpha^*}{\pi C_\alpha^* + 1 - \pi},$$

and the assumption that $\eta \leq \nu'_{\max}(< 1)$ (also $D_\alpha^* \leq \nu'_{\max}$ should follow). Indeed, let

$$f(x) = \frac{\pi x}{\pi x + (1-\pi)}, 0 < x < \nu'_{\max}.$$

Then,

$$g(x) = f^{-1}(x) = \frac{1-\pi}{\pi}\frac{x}{1-x}, 0 < x < \frac{\pi\nu'_{\max}}{\pi\nu'_{\max} + 1 - \pi}.$$

Since $|g'(x)| \leq \frac{\pi}{1-\pi}\left(\frac{\pi\mu'_{\max}+1-\pi}{\pi\nu'_{\max}}\right)^2 =: U$, g is Lipschitz with Lipschitz constant U. Therefore,

$$|\eta - D_\alpha^*| \leq \delta_1 \implies \left|\frac{p_1}{p_0} - C_\alpha^*\right| \leq U\delta_1.$$

This implies

$$\int_{A_1}\left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0 \leq U\delta_1 P_0(A_1) \leq C_0 U \delta_1^{1+\bar{\gamma}},$$

where the second inequality follows from the margin assumption. To bound the integral over A_2, we control the distance between $\hat{D}_\alpha$ and D_α^*. Indeed,

$$\begin{aligned}\alpha &= P_0(\hat{\eta}(X) \geq \hat{D}_\alpha) = P_0(\eta(X) \geq D_\alpha^*) \\ &\geq P_0(\hat{\eta}(X) \geq D_\alpha^* + \delta_1/2).\end{aligned}$$

This implies that $\hat{D}_\alpha \leq D_\alpha^* + \delta_1/2$. So

$$P_0(A_2) \leq P_0(\eta > D_\alpha^* + \delta_1, \hat{\eta} < D_\alpha^* + \delta_1/2) = 0.$$

Therefore,

$$\int_{G^{*c}\cap\hat{G}}\left|\frac{p_1}{p_0} - C_\alpha^*\right| \leq C_0 U \delta_1^{1+\bar{\gamma}}.$$

Similarly bound the integral over $G^* \cap \hat{G}^c$, then

$$\int_{G^* \triangle \hat{G}} \left| \frac{p_1}{p_0} - C_\alpha^* \right| \leq 2C_0 U \delta_1^{1+\bar{\gamma}}.$$

Lemma 2 implies that $\mathbb{P}(\mathcal{E}) \geq 1-\delta$. This concludes the proof. ■

Note that Lemma 2 is a uniform deviation result on the Nadaraya-Watson estimator, and it is the first result of such kind to the best of our knowledge.

4.2 Class 0 Density p_0 Unknown

In this subsection, the assumption of knowledge on p_0 is relaxed. Suppose in addition to the mixed sample $\bar{\mathcal{S}} = \{(X_1, Y_1), \ldots, (X_m, Y_m)\}$, we have access to a class 0 sample $\mathcal{S}_0 = \{X_1^-, \ldots, X_n^-\}$. Moreover, assume that variables in $\mathcal{S}_0$ are independent given $\bar{\mathcal{S}}$. As in the p_0 known case, the notation $\hat{\eta}$ denotes the Nadaraya-Watson estimator based on the sample $\bar{\mathcal{S}}$. Just like $\hat{C}_\alpha$ in Definition 3, we need to define the threshold level $\hat{D}_\alpha$ carefully.

Definition 4 *Fix $\delta \in (0,1)$, for $d_n = 2\sqrt{2\frac{\log(2en)+\log(2/\delta)}{n}}$, let $\hat{D}_\alpha$ be the smallest L such that*

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\left(\hat{\eta}(X_i^-) \geq L\right) \leq \alpha - d_n .$$

Unlike the previous setup where p_0 is known, we now bound the excess type I error.

Proposition 4 *With probability at least $1-\delta$, type I error of the plug-in classifier $\tilde{\phi}$ defined in* (10) *is bounded from above by α, that is,*

$$R_0(\tilde{\phi}) \leq \alpha .$$

The proof is omitted due to similarity to that of Proposition 2. A similar α level detection condition can be formulated can be formulated for the regression function, but we omit it as C_α^* is simply replaced by D_α^*. The next theorem address the excess type II error of $\tilde{\phi}$: $R_1(\tilde{\phi}) - R_1(\phi^*)$.

Theorem 3 *Let $\tilde{\phi} = \mathbb{I}(\hat{\eta} \geq \hat{D}_\alpha)$ be defined as in* (10)*. Assume condition 3 and the regression function η satisfies the level α detection condition for some $\gamma (\geq \bar{\gamma})$. Take the bandwidth $h = (\frac{\log m}{m})^{1/(2\beta+d)}$ in the Nadaraya-Watson estimator $\hat{\eta}$, where the kernel K is β-valid and L'-Lipschitz. Then there exists a positive constant $\bar{C}$, such that for any $\delta \in (0,1)$ and any $m, n \geq 1/\delta$, it holds with probability $1-2\delta$,*

$$R_1(\tilde{\phi}) - R_1(\phi^*) \leq \bar{C}\left[\left(\frac{\log n}{n}\right)^{\min\left(\frac{1}{2}, \frac{1+\bar{\gamma}}{2\gamma}\right)} + \left(\frac{\log m}{m}\right)^{\frac{\beta(1+\bar{\gamma})}{2\beta+d}}\right].$$

Proof Let $G^* = \{\eta < D_\alpha^*\}$ and $\hat{G} = \{\hat{\eta} < \hat{D}_\alpha\}$, then the excess type II error of $\tilde{\phi}$ can be decomposed by

$$P_1(\hat{G}) - P_1(G^*) = \int_{G^* \triangle G^*} \left| \frac{p_1}{p_0} - C_\alpha^* \right| dP_0 + C_\alpha^* + \left[P_0(G^{*c}) - P_0(\hat{G}^c)\right]. \tag{11}$$

Recall that $P_0(G^{*c}) = \alpha$ and $P_0(\hat{G}^c)$ is type I error of $\tilde{\phi}$. By the definition of $\hat{D}_\alpha$, for any positive number l, the following holds for all $n \geq 1$,

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(\hat{\eta}(X_i^-) \geq \hat{D}_\alpha - l) > \alpha - d_n\,,$$

where $d_n = 2\sqrt{2\frac{\log(2en)+\log(2/\delta)}{n}}$. By the same argument as in the proof of Proposition 2, there exists an event $\bar{\mathcal{E}}_l$ regarding the samples $\mathcal{S}_0$, $\tilde{\mathcal{S}}_0$ and $\mathcal{S}_1$ with $\mathbb{P}(\bar{\mathcal{E}}_l) \geq 1-\delta$, such that on this event,

$$P_0\left(\hat{\eta} \geq \hat{D}_\alpha - l\right) \geq \alpha - 2d_n\,.$$

To control the second part of R.H.S. in(11), let $\hat{G}_l = \{\hat{\eta} < \hat{D}_\alpha - l\}$, then

$$P_0(G^{*c}) - P_0(\hat{G}^c) = \inf_{l>0}\left[P_0(G^{*c}) - P_0(\hat{G}_l^c)\right] \leq \alpha - (\alpha - 2d_n) = 2d_n\,, \tag{12}$$

on the event $\bar{\mathcal{E}} := \cap_{l>0}\bar{\mathcal{E}}_l$, and $\mathbb{P}(\bar{\mathcal{E}}) = \lim_{l\to 0}\mathbb{P}(\bar{\mathcal{E}}_l) \geq 1-\delta$. Therefore, it remains to control the first part of R.H.S. in (11). Define an event regarding the sample $\bar{\mathcal{E}}$,

$$\mathcal{E} = \{\|\hat{\eta} - \eta\|_\infty < \delta_1/2\}\,,$$

where $\delta_1 = D_1\sqrt{\frac{\log(3m/\delta)}{mh^d}}$ and D_1 is a constant chosen as in Lemma 2. Lemma 2 implies that $\mathbb{P}(\mathcal{E}) \geq 1-\delta$. We restrict ourselves to $\mathcal{E}\cap\bar{\mathcal{E}}$ for the rest of the proof. Note that the first part of R.H.S. of (11) can be decomposed by

$$\int_{G^*\triangle\hat{G}}\left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0 \quad = \quad \int_{G^{*c}\cap\hat{G}}\left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0 + \int_{G^*\cap\hat{G}^c}\left|\frac{p_1}{p_0} - C_\alpha^*\right| dP_0\,.$$

We will focus the integral over $G^*\cap\hat{G}^c$, as that over $G^{*c}\cap\hat{G}$ can be bounded similarly. Because $|\hat{\eta}-\eta| < \delta_1/2$, for all $l > 0$,

$$\alpha - 2d_n \leq P_0\left(\hat{\eta} \geq \hat{D}_\alpha - l\right) \leq P_0\left(\eta \geq \hat{D}_\alpha - l - \delta_1/2\right)\,.$$

We need to bound $\hat{D}_\alpha$ in terms of D_α^*. This is achieved through the following steps. First, we determine some $c_n > 0$ such that

$$P_0\left(\eta \geq D_\alpha^* + c_n\right) \leq \alpha - 2d_n\,,$$

which follows if the next inequality holds

$$2d_n \leq P_0\left(D_\alpha^* < \eta < D_\alpha^* + c_n\right)\,.$$

By the level α detection condition, it is enough to take $c_n = \left(2d_n/C_1\right)^{1/\gamma}$. Therefore,

$$P_0\left(\eta \geq D_\alpha^* + \left(2d_n/C_1\right)^{1/\gamma}\right) \leq \alpha - 2d_n \leq P_0\left(\eta \geq \hat{D}_\alpha - l - \delta_1/2\right)\,.$$

This implies that

$$\hat{D}_\alpha \leq D_\alpha^* + \left(2d_n/C_1\right)^{1/\gamma} + l + \delta_1/2\,.$$

Since the above holds for all $l > 0$, we have

$$\hat{D}_\alpha \leq D^*_\alpha + (2d_n/C_1)^{1/\gamma} + \delta_1/2\,.$$

We can decompose $G^{*c} \cap \hat{G}$ by

$$G^{*c} \cap \hat{G} = \left\{\eta \geq D^*_\alpha, \hat{\eta} < \hat{D}_\alpha\right\} = A_1 \cap A_2\,,$$

where

$$A_1 = \left\{D^*_\alpha + (2d_n/C_1)^{1/\gamma} + \delta_1 > \eta \geq D^*_\alpha, \hat{\eta} < \hat{D}_\alpha\right\}, \text{ and}$$

$$A_2 = \left\{\eta \geq D^*_\alpha + (2d_n/C_1)^{1/\gamma} + \delta_1, \hat{\eta} < \hat{D}_\alpha\right\}.$$

Let $U := \frac{\pi}{1-\pi}\left(\frac{\pi\mu'_{\max}+1-\pi}{\pi\nu'_{\max}}\right)^2$, through the same derivation as the p_0 known case,

$$|\eta - D^*_\alpha| \leq (2d_n/C_1)^{1/\gamma} + \delta_1 \implies \left|\frac{p_1}{p_0} - C^*_\alpha\right| \leq U\left((2d_n/C_1)^{1/\gamma} + \delta_1\right).$$

By the margin assumption,

$$\int_{A_1}\left|\frac{p_1}{p_0} - C^*_\alpha\right| dP_0 \leq U\left((2d_n/C_1)^{1/\gamma} + \delta_1\right) P_0(A_1) \leq C_0 U\left[(2d_n/C_1)^{1/\gamma} + \delta_1\right]^{1+\bar{\gamma}}.$$

Note that

$$A_2 \subset A_3 := \left\{\eta \geq D^*_\alpha + (2d_n/C_1)^{1/\gamma} + \delta_1, \hat{\eta} < D^*_\alpha + (2d_n/C_1)^{1/\gamma} + \delta_1/2\right\},$$

but $|\eta - \hat{\eta}| < \delta_1/2$ on $\mathcal{E} \cap \bar{\mathcal{E}}$, so $P_0(A_2) = P_0(A_3) = 0$. Therefore,

$$\int_{G^{*c}\cap\hat{G}}\left|\frac{p_1}{p_0} - C^*_\alpha\right| dP_0 \leq C_0 U\left[(2d_n/C_1)^{1/\gamma} + \delta_1\right]^{1+\bar{\gamma}}.$$

Similarly, the integral over $G^* \cap \hat{G}^c$ can be bounded, so we have

$$\int_{G^*\triangle\hat{G}}\left|\frac{p_1}{p_0} - C^*_\alpha\right| dP_0 \leq 2C_0 U\left[(2d_n/C_1)^{1/\gamma} + \delta_1\right]^{1+\bar{\gamma}}. \tag{13}$$

Finally, note that $\mathbb{P}(\mathcal{E} \cap \bar{\mathcal{E}}) \geq 1 - 2\delta$. So (11), (12) and (13) together conclude the proof.

■

In anomaly detection applications, normal samples are considered abundant, that is, $n \gg m$, which implies that $(\frac{\log n}{n})^{\frac{1}{2\beta+d}} \leq (\frac{\log m}{m})^{\frac{1}{2\beta+d}}$. Then the upper bounds for the excess type II errors in Theorem 2 and Theorem 3 are of the same order. Having access to the mixture (contaminated) sample $\bar{\mathcal{S}}$ looks like a weaker condition than having access to a pure anomaly sample. However, this

does not seem to be the case in our settings. The essence is revealed by observing that the density ratio p_1/p_0 and the regression function η play the same role in the oracle NP classifier at level α:

$$\phi^*(x) = \mathbb{I}(p_1/p_0(x) \geq C_\alpha^*) = \mathbb{I}(\eta(x) \geq D_\alpha^*)\,.$$

A plug-in classifier depends upon an estimate of either p_1/p_0 or η. Being able to estimate the anomaly density p_1 is not of particular advantage, because only the ratio p_1/p_0 matters. Strictly speaking, the conditions for the two theorems are not the same, but one advantage of targeting the regression function seems to be that we do not have to split the normal example into two, with one to estimate p_0 and the other to estimate the optimal threshold C_α^*.

Acknowledgments

The author thanks Philippe Rigollet for thought-provoking discussions, and anomalous referees for constructive advice. This research was conducted with generous support at Statlab, Princeton University and Department of Mathematics, MIT.

Appendix A. Technical Lemmas

The appendix includes two important technical lemmas and their proofs. Lemma 1 is a uniform deviation result on kernel density estimators, and Lemma 2 is a uniform division result on Nadaraya-Watson estimators.

Lemma 1 *Let $p \in \mathcal{P}_\Sigma(\beta, L, [-1,1]^d)$ and the kernel K be β-valid and L'-Lipschitz. Denote by $\hat{p}(x) = \frac{1}{nh^d}\sum_{i=1}^n K\left(\frac{X_i - x}{h}\right)$ the kernel density estimator of p based on the i.i.d sample $\{X_1, \ldots, X_n\}$, where h is the bandwidth. For any $\varepsilon \in (0,1)$, if the sample size n is such that $\sqrt{\frac{\log(n/\varepsilon)}{nh^d}} < 1$, it holds*

$$\mathbb{P}(\|\hat{p} - p\|_\infty \geq \delta) \leq \varepsilon\,,$$

where

$$\delta = (32c_2 d + \sqrt{48dc_1})\sqrt{\frac{\log(n/\varepsilon)}{nh^d}} + 2Lc_3 h^\beta + \frac{1}{nh^d}\frac{\sqrt{d}L'}{nh} + (L + \tilde{C}\sum_{1 \leq |s| \leq \lfloor\beta\rfloor}\frac{1}{s!})d^{\beta/2}n^{-2\beta}\,,$$

where $c_1 = \|p\|_\infty \|K\|^2$, $c_2 = \|K\|_\infty + \|p\|_\infty + \int |K||t\|^\beta dt$, $c_3 = \int |K||t\|^\beta dt$, and $\tilde{C}$ is such that $\tilde{C} \geq \sup_{1 \leq |s| \leq \lfloor\beta\rfloor} \sup_{x \in [-1,1]^d} |D^s p(x)|$.

Let $h = \left(\frac{\log n}{n}\right)^{\frac{1}{2\beta+d}}$, it is enough to take $\delta = C\sqrt{\frac{\log(n/\varepsilon)}{nh^d}}$, where

$$C = 32c_2 d + \sqrt{48dc_1} + 2Lc_3 + \sqrt{d}L' + L + \tilde{C}\sum_{1 \leq |s| \leq \lfloor\beta\rfloor}\frac{1}{s!}\,.$$

Proof Divide each coordinate of the hypercube $[-1,1]^d$ into $2M$ equally spaced subintervals. Then $[-1,1]^d$ is subdivided into $(2M)^d$ small hypercubes, with a total of $(2M+1)^d$ vertices. We denote the collection of these vertices by G. Note that for any $\delta > 0$,

$$\mathbb{P}\left(\|\hat{p} - p\|_\infty \geq \delta\right) \leq \mathbb{P}\left(M_1 + M_2 + M_3 \geq \delta\right), \tag{14}$$

where

$$M_1 = \sup_{\|x-x'\| \le \frac{\sqrt{d}}{M}} \frac{1}{nh^d} \left| \sum_{i=1}^{n} \left(K(\frac{X_i - x}{h}) - K(\frac{X_i - x'}{h}) \right) \right|,$$
$$M_2 = \sup_{\|x-x'\| \le \frac{\sqrt{d}}{M}} |p(x) - p(x')|,$$
$$M_3 = \sup_{x \in G} \left| \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{X_i - x}{h}) - p(x) \right|.$$

Note that because K is L'-Lipschitz,

$$M_1 \le \sup_{\|x-x'\| \le \frac{\sqrt{d}}{M}} \frac{1}{nh^d} \sum_{i=1}^{n} \left| \left(K(\frac{X_i - x}{h}) - K(\frac{X_i - x'}{h}) \right) \right| \le \frac{1}{nh^d} \frac{n\sqrt{d}L'}{Mh} = \frac{1}{nh^d} \frac{\sqrt{d}L'}{nh}.$$

To control M_2, note that if $\beta \le 1$,

$$|p(x) - p(x')| = |p(x) - p_{x'}(x)| \le L\|x - x'\|^{\beta}.$$

If $\beta > 1$, p is $\lfloor \beta \rfloor$-times continuously differentiable. In particular, for all s such that $1 \le |s| \le \lfloor \beta \rfloor$, $D^s p$ is continuous. Since $[-1,1]^d$ is compact, there exists a positive constant $\tilde{C}$, such that

$$\sup_{1 \le |s| \le \lfloor \beta \rfloor} \sup_{x \in [-1,1]^d} |D^s p(x)| \le \tilde{C}.$$

Therefore,

$$\begin{aligned} |p(x) - p(x')| &\le |p(x) - p_{x'}(x)| + |p_{x'}(x) - p(x')| \\ &= L\|x - x'\|^{\beta} + \sum_{1 \le |s| \le \lfloor \beta \rfloor} \left| \frac{(x' - x)^s}{s!} D^s p(x') \right| \\ &\le \left(L + \tilde{C} \sum_{1 \le |s| \le \lfloor \beta \rfloor} \frac{1}{s!} \right) \|x - x'\|^{\beta}. \end{aligned}$$

Putting together the $\beta \le 1$ and $\beta > 1$ cases yields $M_2 \le \left(L + \tilde{C} \sum_{1 \le |s| \le \lfloor \beta \rfloor} \frac{1}{s!} \right) d^{\beta/2} n^{-2\beta}$.

Define by $t = \delta - \frac{1}{nh^d} \frac{\sqrt{d}L'}{nh} - \left(L + \tilde{C} \sum_{1 \le |s| \le \lfloor \beta \rfloor} \frac{1}{s!} \right) d^{\beta/2} n^{-2\beta}$. Inequality (14) together with upper bounds on M_1 and M_2, implies that

$$\mathbb{P}\left(\|\hat{p} - p\|_{\infty} \ge \delta \right) \le \mathbb{P}\left(M_3 \ge t \right).$$

Use a union bound to control the tail probability of M_3,

$$\begin{aligned} \mathbb{P}(M_3 \ge t) &\le \sum_{x \in G} \mathbb{P}\left(\left| \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{X_i - x}{h}) - p(x) \right| \ge t \right). \\ &\le 2(2M+1)^d \exp\left(-\frac{h^d n t^2}{8(c_1 + c_2 t/6)} \right), \end{aligned}$$

for $t \geq 2Lc_3h^\beta$. The last inequality relies on the assumptions that $p \in \mathcal{P}_\Sigma(\beta, L, [-1,1]^d)$ and K is β-valid. It essentially follows along the same lines as the proof of Lemma 4.1 in Rigollet and Vert (2009), so we omit the derivation and refer the interested reader to this paper. For a given $\varepsilon \in (0,1)$, we would like to enforce

$$2(2M+1)^d \exp\left(-\frac{h^d n t^2}{8(c_1 + c_2 t/6)}\right) \leq \varepsilon .$$

Because $\log 2 + d\log(2M+1) \leq 6d\log n$, it is sufficient to have

$$6d\log n - \frac{nh^d t^2}{8(c_1 + c_2 t/6)} \leq \log \varepsilon .$$

It is clear there exists a positive t^* such that the above inequality attains equality, and that this inequality holds for all $t \geq t^*$. To get t^*, we restrict ourselves to $t > 0$, so that we have $c_1 + c_2 t/6 > 0$. Then we solve for t^* as the bigger root of a quadratic function in t:

$$t^* = \frac{1}{2nh^d}\left(8c_2 d\log n - \frac{4}{3}\log\varepsilon c_2 + \sqrt{(8c_2 d\log n - \frac{4}{3}\log\varepsilon c_2)^2 - 4nh^d(8c_1\log\varepsilon - 48dc_1\log n)}\right).$$

Observe that for positive a and b, $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$, so it holds

$$\begin{aligned} t^* &\leq \frac{1}{2nh^d}\left(2(8c_2 d\log n - \frac{4}{3}\log\varepsilon c_2) + \sqrt{4nh^d(48dc_1\log n - 8c_1\log\varepsilon)}\right) \\ &\leq 32c_2 d\frac{\log\frac{n}{\varepsilon}}{nh^d} + \sqrt{48dc_1}\sqrt{\frac{\log\frac{n}{\varepsilon}}{nh^d}} \\ &\leq (32c_2 d + \sqrt{48dc_1})\sqrt{\frac{\log\frac{n}{\varepsilon}}{nh^d}}, \end{aligned}$$

in which the last inequality holds for n such that $\sqrt{\frac{\log(n/\varepsilon)}{nh^d}} < 1$. Then we can take

$$\delta = (32c_2 d + \sqrt{48dc_1})\sqrt{\frac{\log\frac{n}{\varepsilon}}{nh^d}} + 2Lc_3 h^\beta + \frac{1}{nh^d}\frac{\sqrt{d}L'}{nh} + \left(L + \tilde{C}\sum_{1\leq|s|\leq\lfloor\beta\rfloor}\frac{1}{s!}\right)d^{\beta/2}n^{-2\beta}.$$

When $h = \left(\frac{\log n}{n}\right)^{\frac{1}{2\beta+d}}$, we have $\sqrt{\frac{\log n}{nh^d}} = h^\beta = \left(\frac{\log n}{n}\right)^{\frac{\beta}{2\beta+d}}$, and $nh > 1$. Also, it is safe to assume that $d^{\beta/2}n^{-2\beta} \leq \sqrt{\log(n/\varepsilon)/(nh^d)}$, Therefore,

$$\begin{aligned} \delta &\leq (32c_2 d + \sqrt{48dc_1})\sqrt{\frac{\log\frac{n}{\varepsilon}}{nh^d}} + 2Lc_3\sqrt{\frac{\log n}{nh^d}} + \sqrt{d}L'\sqrt{\frac{\log n}{nh^d}} + \left(L + \tilde{C}\sum_{1\leq|s|\leq\lfloor\beta\rfloor}\frac{1}{s!}\right)\frac{d^{\beta/2}}{n^{2\beta}} \\ &\leq C\sqrt{\frac{\log\frac{n}{\varepsilon}}{nh^d}}, \end{aligned}$$

where $C = 32c_2 d + \sqrt{48dc_1} + 2Lc_3 + \sqrt{d}L' + L + \tilde{C}\sum_{1\leq|s|\leq\lfloor\beta\rfloor}\frac{1}{s!}$.

■

Lemma 2 *Denote by $\hat{\eta}$ the Nadaraya-Watson estimator of the regression function η based on an i.i.d. sample $\{(X_1,Y_1),\ldots,(X_n,Y_n)\}$. Let p be the marginal density of X, $p \in \mathcal{P}_\Sigma(\beta,L,[-1,1]^d)$, $f = \eta \cdot p \in \mathcal{P}_\Sigma(\beta,L,[-1,1]^d)$, and the kernel K be β-valid and L'-Lipschitz. Moreover, assume $p \geq \mu'_{\min}(>0)$ and the sample size n is such that $\sqrt{\frac{\log(n/\varepsilon)}{nh^d}} < 1$. Then for any $\varepsilon > 0$,*

$$\mathbb{P}(\|\hat{\eta}-\eta\|_\infty \geq \delta) \leq 3\varepsilon,$$

for

$$\begin{aligned}\delta &= \frac{1}{\mu'_{\min}-\delta'}\left(\delta' + (32d\|K\|_\infty + \sqrt{12d\|K\|^2\|p\|_\infty})\sqrt{\frac{\log(n/\varepsilon)}{nh^d}} + (c_4+c_5)h^\beta\right)\\ &+ \frac{1}{\mu'_{\min}-\delta'}\left(\frac{1}{nh^d}\frac{\sqrt{d}L'}{nh} + \left(L + \tilde{C}_1 \sum_{1\leq|s|\leq\lfloor\beta\rfloor}\frac{1}{s!}\right)\frac{d^{\beta/2}}{n^{2\beta}}\right),\end{aligned}$$

where δ' is the same as δ in Lemma 1, $c_4 = \frac{\|p\|_\infty L}{\mu'_{\min}}\left(1+\frac{\|f\|_\infty}{\mu'_{\min}}\right)\left(\int|K(z)|\cdot\|z\|^\beta dz\right)$ and $c_5 = L\int|K(z)|\cdot\|z\|^\beta dz$, and $\tilde{C}_1$ is such that $\tilde{C}_1 \geq \sup_{1\leq|s|\leq\lfloor\beta\rfloor}\sup_{x\in[-1,1]^d}|D^s p(x)|$.

In particular, when $h = (\frac{\log n}{n})^{\frac{1}{2\beta+d}}$, there exists some positive D, such that we can take $\delta = D\sqrt{\frac{\log(n/\varepsilon)}{nh^d}}$.

Proof Recall that $\hat{p}(x) = \frac{1}{nh^d}\sum_{i=1}^n K(\frac{X_i-x}{h})$ and $\hat{f}(x) = \frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i-x}{h})$, so

$$|\hat{\eta}-\eta| = \left|\frac{\hat{f}}{\hat{p}}-\frac{f}{p}\right| \leq \left|\frac{f}{\hat{p}}-\frac{f}{p}\right| + \left|\frac{\hat{f}}{\hat{p}}-\frac{f}{\hat{p}}\right| = |f|\frac{|\hat{p}-p|}{|\hat{p}||p|} + \frac{1}{|\hat{p}|}|\hat{f}-f|.$$

This implies that

$$\mathbb{P}(\|\hat{\eta}-\eta\|_\infty \geq \delta) \leq \mathbb{P}\left(\left\||f|\frac{|\hat{p}-p|}{|\hat{p}||p|}\right\|_\infty + \left\|\frac{1}{|\hat{p}|}|\hat{f}-f|\right\|_\infty \geq \delta\right).$$

Therefore, to claim $\mathbb{P}(\|\hat{\eta}-\eta\|_\infty \geq \delta) \leq 3\varepsilon$, it is enough to show that for δ_1 and δ_2 such that $\delta = \delta_1+\delta_2$, it holds

$$\text{i) } \mathbb{P}\left(\left\||f|\frac{|\hat{p}-p|}{|\hat{p}||p|}\right\|_\infty \geq \delta_1\right) \leq \varepsilon, \text{ and ii) } \mathbb{P}\left(\left\|\frac{1}{|\hat{p}|}|\hat{f}-f|\right\|_\infty \geq \delta_2\right) \leq 2\varepsilon, \tag{15}$$

where the quantities δ_1 and δ_2 will be specified later.

i). We prove the first inequality in (15). Because $\eta \leq 1$ and $\eta(x) = f(x)/p(x)$,

$$\mathbb{P}\left(\left\||f|\frac{|\hat{p}-p|}{|\hat{p}||p|}\right\|_\infty \geq \delta_1\right) \leq \mathbb{P}\left(\left\|\frac{\hat{p}-p}{\hat{p}}\right\|_\infty \geq \delta_1\right).$$

Denote an event regarding the sample $\mathcal{E} = \{\|\hat{p}-p\|_\infty < \delta'\}$, where δ' is the same as δ in Lemma 1. So by Lemma 1, $\mathbb{P}(\mathcal{E}) > 1-\varepsilon$. Moreover,

$$\begin{aligned}&\mathbb{P}(\|(\hat{p}-p)/\hat{p}\|_\infty \geq \delta_1)\\ \leq\ &\mathbb{P}\left(\|(\hat{p}-p)/\hat{p}\|_\infty \geq \delta_1, \|\hat{p}-p\|_\infty \geq \delta'\right) + \mathbb{P}\left(\|(\hat{p}-p)/\hat{p}\|_\infty \geq \delta_1, \|\hat{p}-p\|_\infty < \delta'\right)\\ \leq\ &\mathbb{P}(\|\hat{p}-p\|_\infty \geq \delta') + \mathbb{P}\left(\|\hat{p}-p\|_\infty \geq \delta_1(\mu'_{\min}-\delta'), \|\hat{p}-p\|_\infty < \delta'\right).\end{aligned}$$

Take $\delta_1 = \frac{\delta'}{\mu'_{\min} - \delta'}$, then we have $\delta' = \delta_1(\mu'_{\min} - \delta')$. So

$$\mathbb{P}\left(\left\| |f| \frac{|\hat{p} - p|}{|\hat{p}||p|} \right\|_\infty \geq \delta_1 \right) \leq \mathbb{P}(\|\hat{p} - p\|_\infty \geq \delta') \leq \varepsilon .$$

ii). Now we prove the second inequality in (15). Note that

$$\begin{aligned} & \mathbb{P}\left(\left\| \frac{\hat{f} - f}{\hat{p}} \right\|_\infty \geq \delta_2 \right) \\ = \; & \mathbb{P}\left(\left\| \frac{\hat{f} - f}{\hat{p}} \right\|_\infty \geq \delta_2, \|\hat{p} - p\|_\infty \geq \delta' \right) + \mathbb{P}\left(\left\| \frac{\hat{f} - f}{\hat{p}} \right\|_\infty \geq \delta_2, \|\hat{p} - p\|_\infty < \delta' \right) \\ \leq \; & \mathbb{P}(\|\hat{p} - p\|_\infty \geq \delta') + \mathbb{P}\left(\|\hat{f} - f\|_\infty \geq \delta_2(\mu'_{\min} - \delta') \right) \\ = \; & \varepsilon + \mathbb{P}\left(\|\hat{f} - f\|_\infty \geq \delta_2(\mu'_{\min} - \delta') \right) . \end{aligned}$$

Therefore, to bound the tail probability of $\|(\hat{f} - f)/\hat{p}\|_\infty$, it remains to show

$$\mathbb{P}(\|\hat{f} - f\|_\infty \geq \delta_2(\mu'_{\min} - \delta')) \leq \varepsilon .$$

Let G be the same collection of vertices of sub-cubes in $[-1,1]^d$ as in the proof of Lemma 1, and denote by $M = n^2$. Note that for every $\delta_3 > 0$, it holds

$$\mathbb{P}(\|\hat{f} - f\|_\infty \geq \delta_3) \quad \leq \quad \mathbb{P}(M_1 + M_2 + M_3 \geq \delta_3) ,$$

where

$$\begin{aligned} M_1 &= \sup_{\|x - x'\| \leq \frac{\sqrt{d}}{M}} |\hat{f}(x) - \hat{f}(x')| , \\ M_2 &= \sup_{\|x - x'\| \leq \frac{\sqrt{d}}{M}} |f(x) - f(x')| , \\ M_3 &= \sup_{x \in G} |\hat{f}(x) - f(x)| . \end{aligned}$$

The quantity M_1 can be controlled as follows:

$$\begin{aligned} M_1 \; &= \; \sup_{\|x - x'\| \leq \frac{\sqrt{d}}{M}} \left| \frac{1}{nh^d} \sum_{i=1}^n Y_i K(\frac{X_i - x}{h}) - \frac{1}{nh^d} \sum_{i=1}^n Y_i K(\frac{X_i - x'}{h}) \right| \\ &\leq \; \sup_{\|x - x'\| \leq \frac{\sqrt{d}}{M}} \frac{1}{nh^d} \sum_{i=1}^n \left| K(\frac{X_i - x}{h}) - K(\frac{X_i - x'}{h}) \right| \\ &\leq \; \frac{1}{nh^d} \frac{n\sqrt{d}L'}{Mh} = \frac{1}{nh^d} \frac{\sqrt{d}L'}{nh} . \end{aligned}$$

The quantity M_2 can be controlled similarly as its counterpart in proof for Lemma 1,

$$M_2 \leq \left(L + \tilde{C}_1 \sum_{1 \leq |s| \leq \lfloor \beta \rfloor} \frac{1}{s!} \right) d^{\frac{\beta}{2}} n^{-2\beta} .$$

Let $t = \delta_3 - \frac{1}{nh^d}\frac{\sqrt{d}L'}{nh} - \left(L + \tilde{C}_1 \sum_{1\le|s|\le\lfloor\beta\rfloor} \frac{1}{s!}\right) d^{\frac{\beta}{2}} n^{-2\beta}$, then

$$\mathbb{P}(\|\hat{f} - f\|_\infty \ge \delta_3) \le \mathbb{P}(M_3 \ge t)\,.$$

Use a union bound to control the tail probability of M_3:

$$\mathbb{P}(M_3 \ge t) \le \sum_{x\in G} \mathbb{P}\left(|\hat{f}(x) - f(x)| \ge t\right)\,.$$

For each fixed $x \in G$,

$$\begin{aligned}
\hat{f}(x) - f(x) &= \frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i - x}{h}) - \eta(x)\cdot p(x) \\
&= \frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i - x}{h}) - \mathbb{E}\left[\frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i - x}{h})\right] \\
&+ \mathbb{E}\left[\frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i - x}{h})\right] - \eta(x)\cdot p(x) \\
&= B_1(x) + B_2(x)\,,
\end{aligned}$$

where

$$\begin{aligned}
B_1(x) &= \frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i - x}{h}) - \mathbb{E}\left[\frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i - x}{h})\right]\,, \\
B_2(x) &= \mathbb{E}\left[\frac{1}{nh^d}\sum_{i=1}^n Y_i K(\frac{X_i - x}{h})\right] - \eta(x)\cdot p(x)\,.
\end{aligned}$$

This implies that

$$\mathbb{P}(M_1 \ge t) \le \sum_{x\in G} \mathbb{P}(|B_1(x)| + |B_2(x)| \ge t)\,.$$

The tail probability of $|B_1(x)|$ is controlled by invoking the Bernstein's inequality. Denote by $Z_i = Z_i(x) = \frac{1}{h^d} Y_i K(\frac{X_i - x}{h}) - \mathbb{E}\left[\frac{1}{h^d} Y_i K(\frac{X_i - x}{h})\right]$. It is clear that $\mathbb{E}(Z_i) = 0$, $|Z_i| \le 2\|K\|_\infty h^{-d}$. Moreover,

$$Var(Z_i) \le \mathbb{E}\left(h^{-2d} K^2(\frac{X_i - x}{h})\right) = \int h^{-d} K^2(y) p(x + yh) dy \le \|K\|^2 \|p\|_\infty h^{-d}\,.$$

Therefore for any $t_1 > 0$,

$$\begin{aligned}
&\sum_{x\in G} \mathbb{P}\left(|B_1(x)| \ge t_1\right) = \sum_{x\in G} \mathbb{P}\left(\frac{1}{n}|\sum_{i=1}^n Z_i| \ge t_1\right) \\
&\le 2(2M+1)^d \exp\left(-\frac{nt_1^2}{2\|K\|^2\|p\|_\infty h^{-d} + 4/3\|K\|_\infty h^{-d} t_1}\right)\,.
\end{aligned}$$

To have the last display bounded from above by $\varepsilon \in (0,1)$, we recycle the arguments in the proof for Lemma 1 to find out that t_1 should be greater than or equal to

$$t_1^* = \left(32d\|K\|_\infty + \sqrt{12d\|K\|^2\|p\|_\infty}\right)\sqrt{\frac{\log(n/\varepsilon)}{nh^d}},$$

provided the sample size n is such that $\sqrt{\frac{\log(n/\varepsilon)}{nh^d}} < 1$.

Decompose $B_2(x)$ into two parts,

$$\begin{aligned} B_2(x) &= \left\{\mathbb{E}\left[\tfrac{1}{nh^d}\textstyle\sum_{i=1}^n K(\tfrac{X_i-x}{h})\eta(X_i)\right] - \mathbb{E}[\hat{p}(x)\eta(x)]\right\} \\ &\quad + \left\{\mathbb{E}[\hat{p}(x)\eta(x)] - p(x)\eta(x)\right\}. \end{aligned}$$

Note that

$$\begin{aligned} &\left|\mathbb{E}\left[\frac{1}{nh^d}\sum_{i=1}^n K(\frac{X_i-x}{h})\eta(X_i)\right] - \mathbb{E}[\hat{p}(x)\eta(x)]\right| \\ &= \left|\int \frac{1}{h^d}K(\frac{y-x}{h})(\eta(y)-\eta(x))p(y)dy\right| \\ &= \left|\int K(z)[\eta(x+hz)-\eta(x)]p(x+hz)dz\right| \\ &\le \|p\|_\infty \int |K(z)|\cdot|\eta(x+hz)-\eta(x)|dz. \end{aligned} \tag{16}$$

Note that

$$\begin{aligned} |\eta(x+hz)-\eta(x)| &= \left|\frac{f(x+hz)}{p(x+hz)} - \frac{f(x)}{p(x)}\right| \\ &\le \frac{1}{\mu'_{\min}}\left|f(x+hz) - f(x)\frac{p(x+hz)}{p(x)}\right|. \end{aligned}$$

It follows from $p \in \mathcal{P}_\Sigma(L,\beta,[-1,1]^d)$ that

$$\left|\frac{p(x+hz)}{p(x)} - 1\right| \le \frac{L\|z\|^\beta h^\beta}{\mu'_{\min}}.$$

Therefore,

$$\begin{aligned} &|\eta(x+hz)-\eta(x)| \\ &\le \frac{1}{\mu'_{\min}}\left(|f(x+hz)-f(x)| + |f(x)|\cdot\frac{L}{\mu'_{\min}}\|z\|^\beta h^\beta\right) \\ &\le \left(1 + \frac{\|f\|_\infty}{\mu'_{\min}}\right)\frac{L}{\mu'_{\min}}\|z\|^\beta h^\beta, \end{aligned}$$

where the last inequality follows from $f \in \mathcal{P}_\Sigma(\beta,L,[-1,1]^d)$. The above inequality together with (16) implies that

$$\left|\mathbb{E}\left[\frac{1}{nh^d}\sum_{i=1}^n K(\frac{X_i-x}{h})\eta(X_i)\right] - \mathbb{E}[\hat{p}(x)\cdot\eta(x)]\right| \le c_4 h^\beta,$$

where $c_4 = \frac{\|p\|_\infty L}{\mu'_{\min}} \left(1 + \frac{\|f\|_\infty}{\mu'_{\min}}\right) \left(\int |K(z)| \cdot \|z\|^\beta dz\right)$.

Now we control the second part of $B_2(x)$. Because $p \in \mathcal{P}_\Sigma(\beta, L, [-1,1]^d)$, we have via similar lines to the proof of Lemma 4.1 in Rigollet and Vert (2009),

$$|\mathbb{E}[\hat{p}(x)\eta(x)] - p(x)\eta(x)| \leq |\eta(x)| \cdot |\mathbb{E}\hat{p}(x) - p(x)| \leq c_5 h^\beta,$$

where $c_5 = L\int |K(z)| \cdot \|z\|^\beta dz$. Therefore,

$$|B_2(x)| \leq (c_4 + c_5) h^\beta.$$

Taking $\tilde{t} = (32d\|K\|_\infty + \sqrt{12d\|K\|^2\|p\|_\infty})\sqrt{\frac{\log(n/\varepsilon)}{nh^d}} + (c_4 + c_5)h^\beta$, and $\delta_3 = \tilde{t} + \frac{1}{nh^d}\frac{\sqrt{d}L'}{nh} + \left(L + \tilde{C}_1 \sum_{1 \leq |s| \leq \lfloor\beta\rfloor} \frac{1}{s!}\right) d^{\beta/2} n^{-2\beta}$, we have

$$\mathbb{P}(\|\hat{f} - f\|_\infty \geq \delta_3) \leq \mathbb{P}(M_1 \geq \tilde{t}) \leq \sum_{x \in G} \mathbb{P}\left(|B_1(x)| \geq \tilde{t} - (c_4 + c_5)h^\beta\right) \leq \varepsilon.$$

Take $\delta_2 = \frac{\delta_3}{\mu'_{\min} - \delta'}$, we have $\mathbb{P}(\|\hat{f} - f\|_\infty \geq \delta_2(\mu'_{\min} - \delta')) \leq \varepsilon$.

To conclude, part i) and part ii) in (15) together close the proof.

■

References

M. Agyemang, K. Barker, and R. Alhajj. A comprehensive survey of numeric and symbolic outlier mining techniques. *Intelligent Data Analysis*, 6:521–538, 2006.

J. Audibert and A. Tsybakov. Fast learning rates for plug-in classifiers under the margin condition. *Annals of Statistics*, 35:608–633, 2007.

G. Blanchard, G. Lee, and G. Scott. Semi-supervised novelty detection. *Journal of Machine Learning Research*, 11:2973–3009, 2010.

A. Cannon, J. Howse, D. Hush, and C. Scovel. Learning with the neyman-pearson and min-max criteria. *Technical Report LA-UR-02-2951*, 2002.

D. Casasent and X. Chen. Radial basis function neural networks for nonlinear fisher discrimination and neyman-pearson classification. *Neural Networks*, 16(5-6):529 – 535, 2003.

V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 09:1–72, 2009.

C. Elkan. The foundations of cost-sensitive learning. *In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 973–978, 2001.

E. Giné, V. Koltchinskii, and L. Sakhanenko. Kernel density estimators: convergence in distribution for weighted sup norms. *Probability Theory and Related Fields*, 130:167–198, 2004.

M. Han, D. Chen, and Z. Sun. Analysis to Neyman-Pearson classification with convex loss function. *Analysis in Theory and Applications*, 24(1):18–28, 2008.

V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Rivew*, 2:85–126, 2004.

E. L. Lehmann and J. P. Romano. *Testing Statistical Hypotheses*. Springer Texts in Statistics. Springer, New York, third edition, 2005. ISBN 0-387-98864-5.

J. Lei, A. Rinaldo, and L. Wasserman. A conformal prediction approach to explore functional data. *Annals of Mathematics and Artificial Intelligence*, 2013.

O. Lepski. Multivariate density estimation under sup-norm loss: oracle approach, adaptation and independence structure. *Annals of Statistics*, 41(2):1005–1034, 2013.

E. Mammen and A.B. Tsybakov. Smooth discrimination analysis. *Annals of Statistics*, 27:1808–1829, 1999.

M. Markou and S. Singh. Novelty detection: a review-part 1: statistical approahces. *Signal Processing*, 12:2481–2497, 2003a.

M. Markou and S. Singh. Novelty detection: a review-part 2: network-based approahces. *Signal Processing*, 12:2499–2521, 2003b.

A. Patcha and J.M. Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 12:3448–3470, 2007.

W. Polonik. Measuring mass concentrations and estimating density contour clusters-an excess mass approach. *Annals of Statistics*, 23:855–881, 1995.

P. Rigollet and X. Tong. Neyman-pearson classification, convexity and stochastic constraints. *Journal of Machine Learning Research*, 12:2831–2855, 2011.

P. Rigollet and R. Vert. Optimal rates for plug-in estimators of density level sets. *Bernoulli*, 15(4): 1154–1178, 2009.

C. Scott. Comparison and design of neyman-pearson classifiers. Unpublished, 2005.

C. Scott. Performance measures for Neyman-Pearson classification. *IEEE Transactions on Information Theory*, 53(8):2852–2863, 2007.

C. Scott and R. Nowak. A neyman-pearson approach to statistical learning. *IEEE Transactions on Information Theory*, 51(11):3806–3819, 2005.

B. Tarigan and S. van de Geer. Classifiers of support vector machine type with l1 complexity regularization. *Bernoulli*, 12:1045–1076, 2006.

A. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 32: 135–166, 2004.

A. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

A. Tsybakov and S. van de Geer. Square root penalty: Adaptation to the margin in classification and in edge estimation. *Annals of Statistics*, 33:1203–1224, 2005.

D. Wied and R. Weiβbach. Consistency of the kernel density estimator: a survey. *Statistical Papers*, 53(1):1–21, 2010.

Y. Yang. Minimax nonparametric classification-part i: rates of convergence. *IEEE Transaction Information Theory*, 45:2271–2284, 1999.

B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. *IEEE International Conference on Data Mining*, page 435, 2003.

Journal of Machine Learning Research 14 (2013) 3041-3071 Submitted 7/12; Revised 4/13; Published 10/13

Experiment Selection for Causal Discovery

Antti Hyttinen ANTTI.HYTTINEN@HELSINKI.FI
Helsinki Institute for Information Technology
Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Frederick Eberhardt FDE@CALTECH.EDU
Philosophy, Division of the Humanities and Social Sciences
MC 101-40
California Institute of Technology
Pasadena, CA 91125, USA

Patrik O. Hoyer PATRIK.HOYER@HELSINKI.FI
Helsinki Institute for Information Technology
Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Editor: David Maxwell Chickering

Abstract

Randomized controlled experiments are often described as the most reliable tool available to scientists for discovering causal relationships among quantities of interest. However, it is often unclear how many and which different experiments are needed to identify the full (possibly cyclic) causal structure among some given (possibly causally insufficient) set of variables. Recent results in the causal discovery literature have explored various identifiability criteria that depend on the assumptions one is able to make about the underlying causal process, but these criteria are not directly constructive for selecting the optimal set of experiments. Fortunately, many of the needed constructions already exist in the combinatorics literature, albeit under terminology which is unfamiliar to most of the causal discovery community. In this paper we translate the theoretical results and apply them to the concrete problem of experiment selection. For a variety of settings we give explicit constructions of the optimal set of experiments and adapt some of the general combinatorics results to answer questions relating to the problem of experiment selection.

Keywords: causality, randomized experiments, experiment selection, separating systems, completely separating systems, cut-coverings

1. Introduction

In a variety of scientific fields one of the main goals is to understand the causal relationships among some variables of interest. In most cases empirical data is key to discovering and verifying such relationships. While there has been much work on inference principles and algorithms for discovering causal relationships from purely 'passive observational' data, largely based on the seminal work

of Spirtes et al. (1993) and Pearl (2000), randomized controlled experiments (Fisher, 1935) still often constitute the tool of choice for inferring causal relationships. In the more recent literature on causal discovery, randomized experiments, in combination with novel inference principles, play an increasingly prominent role (Cooper and Yoo, 1999; Tong and Koller, 2001; Murphy, 2001; Eaton and Murphy, 2007; Meganck et al., 2005; Eberhardt et al., 2005; He and Geng, 2008; Hyttinen et al., 2010, 2011; Claassen and Heskes, 2010). Thus, given a set of assumptions one is willing to make in a test setting, questions concerning the optimal choices of the manipulations arise. What sequence of experiments identifies the underlying causal structure most efficiently? Or, given some background knowledge, how can one select an experiment that maximizes (in some to be defined sense) the insight one can expect to gain?

In our work (Eberhardt, 2007; Eberhardt et al., 2010; Hyttinen et al., 2011, 2012a,b) we found that many of these questions concerning the optimal selection of experiments are equivalent to graph-theoretic or combinatoric problems for which, in several cases, there exist solutions in the mathematics literature. Generally these solutions are couched in a terminology that is neither common in the literature on causal discovery, nor obvious for its connections to the problems in causal discovery. The present article is intended to bridge this terminological gap, both to indicate which problems of experiment selection already have formal solutions, and to provide explicit procedures for the construction of optimal sets of experiments.[1] It gives rise to new problems that (to our knowledge) are still open and may benefit from the exchange of research in causal discovery on the one hand, and the field of combinatorics and graph theory on the other.

2. Causal Models, Experiments, and Identifiability

We consider causal models which represent the relevant causal structure by a directed graph $G = (\mathcal{V}, \mathcal{D})$, where $\mathcal{V}$ is the set of variables under consideration, and $\mathcal{D} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of directed edges among the variables. A directed edge from $x_i \in \mathcal{V}$ to $x_j \in \mathcal{V}$ represents a direct causal influence of x_i on x_j, with respect to the full set of variables in $\mathcal{V}$ (Spirtes et al., 1993; Pearl, 2000). In addition to the graph G, a fully specified causal model also needs to describe the causal processes that determine the value of each variable given its direct causes. Typically this is achieved either by using conditional probability distributions (in the 'causal Bayes nets' framework) or stochastic functional relationships (for 'structural equation models' (SEMs)).

In addition to describing the system in its 'natural' or 'passive observational' state, a causal model also gives a precise definition of how the system behaves under manipulations. Specifically, consider an intervention that sets (that is, forces) a given variable $x_i \in \mathcal{V}$ to some value randomly chosen by the experimenter. Such a "surgical" intervention corresponds to deleting all arcs pointing *into* x_i (leaving all outgoing arcs, and any other arcs in the model, unaffected), and disregarding the specific process by which x_i normally acquires its value.[2] The resulting graph is known as the 'manipulated' graph corresponding to this intervention. If, in an experiment, the values of *several* variables are set by the experimenter, any arcs into any of those variables are deleted. In this way, a

1. The different constructions and bounds presented in the paper are implemented in a code package at: `http://www.cs.helsinki.fi/group/neuroinf/nonparam/`.

2. In Spirtes et al. (1993) this is how manipulations are defined, which are then combined with the Markov assumption to yield the Manipulation Theorem that specifies formally the relation between the passive observational and a manipulated model. In Pearl (2000) the relation is formally specified using the *do*-operator in combination with the modularity assumption. For our purposes, either connection suffices.

causal model provides a concrete prediction for the behaviour of the system under any experimental conditions.

The problem of causal discovery is to infer (to the fullest extent possible) the underlying causal model, from sample data generated by the model. The data can come either from a passive observational setting (no manipulations performed by the researcher) or from one or more randomized experiments, each of which (repeatedly) sets some subset of the variables to values determined purely by chance, while simultaneously measuring the remaining variables. We define an experiment $\mathcal{E} = (\mathcal{I}, \mathcal{U})$ as a partition of the variable set $\mathcal{V}$ into two mutually exclusive and collectively exhaustive sets $\mathcal{I}$ and $\mathcal{U}$, where $\mathcal{I} \subseteq \mathcal{V}$ represents the variables that are intervened on (randomized) in experiment $\mathcal{E}$, and $\mathcal{U} = \mathcal{V} \setminus \mathcal{I}$ represents the remaining variables, all of which are passively observed in this experiment. We will not consider the specific distributions employed to randomize the intervened variables, except to require that the distribution is positive over all combinations of values of the intervened variables. Note that the identifiability results mentioned below apply when the variables simultaneously intervened on in one experiment are randomized independently of one another.[3]

The extent to which the underlying causal model can be inferred then depends not only on the amount of data available (number of samples) but fundamentally also on the details of what experiments are available and what assumptions on the underlying model one can safely make. In what follows, we only consider model *identifiability*,[4] that is, we disregard sample size and only examine the settings under which models can be learned in the large sample limit. The identifiability results we consider build on causal discovery procedures that make one or more of the following standard assumptions on the underlying model:

acyclicity The graph G is often assumed to be *acyclic*, that is, there exists no directed path from a node back to itself. This assumption is useful for causal discovery because finding that x causes y allows us to deduce that y does not cause x.

causal sufficiency In many cases only a subset of the variables involved in the underlying data generating process are measured. Even if some variables are unobserved, a causal model is said to be *causally sufficient* if there are no unobserved common causes of the observed variables. Unobserved common causes are typically troublesome because they bring about a dependence between two observed variables that is not due to any actual causal process among the observed variables.

faithfulness Many causal discovery procedures use independence tests as a primary tool for inferring the structure of the underlying graph. Such inferences are correct in the limit if the distribution generated by the model is *faithful* to the graph structure, that is, all independencies in distribution are consequences of the graph structure rather than the specific parameter values defining the quantitative relationships between the variables. Under faithfulness, perturbing the parameters defining the quantitative relationships will not break any of the observed independencies between the variables in the distribution.

parametric form Some discovery methods rely on the quantitative causal relations between the variables being restricted to a particular (simple or smooth) *parametric form*. The most com-

3. For linear cyclic models this assumption can be relaxed; see Lemma 5 in Hyttinen et al. (2012b).

4. In contrast, Shpitser and Pearl (2006) and Bareinboim and Pearl (2012) consider the identifiability of single causal effects presuming the knowledge of the true causal structure.

mon such assumption is linearity: the value of each variable is given by a linear sum of the values of its parents plus a stochastic error term.

It is well known that even when one makes *all* of the above assumptions (using only linearity as a parametric restriction), the true causal structure is in general underdetermined given only passive observational data, but can be identified using experiments. We can ask more generally: Under what combination of assumptions and conditions on a set of K experiments $\{\mathcal{E}_1,\ldots,\mathcal{E}_K\}$ is the underlying causal structure identified?

If a total of n observed variables is considered, it should come as no surprise that a set of $K = n$ randomized experiments, each of which intervenes on all but one of the variables, is in general sufficient to uniquely identify the graph G that represents the causal structure among the n variables. In each such experiment we can test which of the $n-1$ other variables are direct causes of the one non-intervened variable. A natural question is whether the full identification of G can be achieved with other sets of experiments, under various combinations of the above assumptions. In particular, we can ask whether identification can be reached with fewer than n experiments, or with experiments that only involve simultaneous interventions on many fewer than $n-1$ variables in each experiment.

Figure 1 provides an example with three variables. Suppose the true causal structure is the chain $x \leftarrow y \leftarrow z$, as shown in graph (i). Assuming only faithfulness *or* linearity, the three experiments intervening on two variables each are sufficient to uniquely identify the true causal graph. In particular, when intervening on both x and y, as illustrated in graph (ii), both of the edges in the true model are cut, and z is independent of both of the intervened variables, indicating that the edges $x \rightarrow z$ and $y \rightarrow z$ are both absent. On the other hand, when intervening on x and z, as shown in graph (iii), a dependence is found between y and z, indicating that $y \leftarrow z$ must be present, while the independence between x and y rules out the edge $x \rightarrow y$. Similar considerations apply when intervening on y and z, illustrated in graph (iv). Together, all potential edges in the model are established as either present or absent, and hence the full causal structure is identified. Note that this inference is possible without assuming causal sufficiency or acyclicity.

Assuming causal sufficiency, acyclicity and faithfulness, fewer experiments are needed: If one had started with an experiment only intervening on x, then a second single intervention experiment on y would be sufficient for unique identifiability. This is because the first experiment, illustrated in graph (v), rules out the edges $x \rightarrow y$ and $x \rightarrow z$, but also establishes, due to the statistical dependence, that y and z are connected by an edge (whose orientation is not yet known). Intervening on y next, as shown in graph (vi), establishes the edge $x \leftarrow y$, and the absence of the edge $y \rightarrow z$, allowing us to conclude (using (v)) that $y \leftarrow z$ must be present in the true graph. Finally, x and z are independent in this second experiment, ruling out the edge $x \leftarrow z$. Thus, the true causal structure is identified. If one had been *lucky* to start with an intervention on z then it turns out that one could have identified the true causal graph in this single experiment. But that single experiment would, of course, have been insufficient if in fact the causal chain had been oriented in the opposite direction.

The example illustrates the sensitivity of the identifiability results to the model space assumptions. However, recent research has shown that, in several different settings (described explicitly below), identification hinges on the set of experiments satisfying some relatively simple conditions. Specifically, consider the following conditions:

Definition 1 (Unordered Pair Condition) *A set of experiments $\{\mathcal{E}_1,\ldots,\mathcal{E}_K\}$ satisfies the unordered pair condition for an unordered pair of variables $\{x_i,x_j\} \subseteq \mathcal{V}$ whenever there is an experiment*

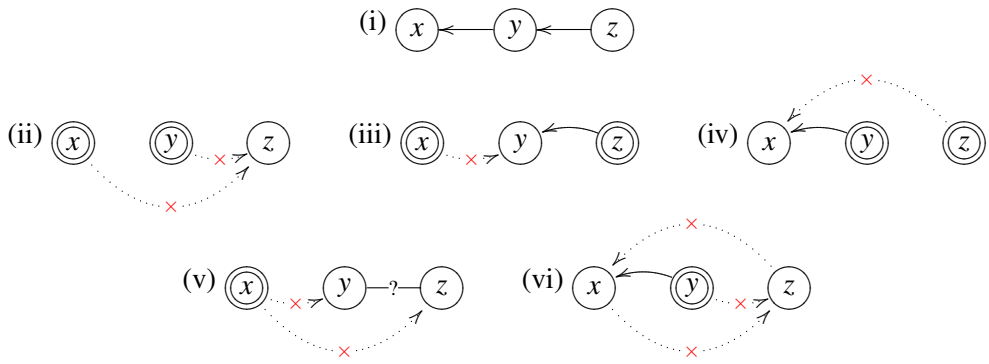


Figure 1: Graph (i) shows the true data generating structure. Graphs (ii-vi) show the possible inferences about the causal structure that can be made from experiments intervening on two variables simultaneously (ii-iv), or intervening on a single variable (v-vi). Variables that are circled twice are intervened on in the corresponding experiment. Edges determined to be present are solid, edges determined to be absent are dotted and crossed out (×). The solid line with a question mark denotes an edge determined to be present but whose orientation is unknown. See the text for a description of the background assumptions that support these inferences.

$\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ in $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ such that $x_i \in \mathcal{J}_k$ (x_i is intervened on) and $x_j \in \mathcal{U}_k$ (x_j is passively observed), or $x_j \in \mathcal{J}_k$ (x_j is intervened on) and $x_i \in \mathcal{U}_k$ (x_i is passively observed).

Definition 2 (Ordered Pair Condition) *A set of experiments $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ satisfies the ordered pair condition for an ordered pair of variables $(x_i, x_j) \in \mathcal{V} \times \mathcal{V}$ (with $x_i \neq x_j$) whenever there is an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ in $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ such that $x_i \in \mathcal{J}_k$ (x_i is intervened on) and $x_j \in \mathcal{U}_k$ (x_j is passively observed).*

Definition 3 (Covariance Condition) *A set of experiments $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ satisfies the covariance condition for an unordered pair of variables $\{x_i, x_j\} \subseteq \mathcal{V}$ whenever there is an experiment $\mathcal{E}_k = (\mathcal{J}_k, \mathcal{U}_k)$ in $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ such that $x_i \in \mathcal{U}_k$ and $x_j \in \mathcal{U}_k$, that is, both variables are passively observed.*

The above conditions have been shown to underlie the following identifiability results: Assuming causal sufficiency, acyclicity and faithfulness, a set of experiments uniquely identifies the causal structure of a causal Bayes net if and only if for any two variables $x_i, x_j \in \mathcal{V}$ one of the following is true: (i) the ordered pair condition holds for the ordered pairs (x_i, x_j) and (x_j, x_i), or (ii) the unordered pair condition and the covariance condition hold for the unordered pair $\{x_i, x_j\}$ (Eberhardt, 2007). Note that the 'only if' part is a worst-case result: For any set of experiments that does not satisfy the above requirement, there exists a causal graph such that the structure is not identified with these experiments. Since a single passive observation—a so-called *null*-experiment, as $\mathcal{J} = \emptyset$—satisfies the covariance condition for all pairs of variables, under the stated assumptions the main challenge is to find experiments that satisfy the *unordered pair condition* for every pair of variables.

Without causal sufficiency, acyclicity *or* faithfulness, but assuming a linear data generating model, a set of experiments uniquely identifies[5] the causal structure among the observed variables of a linear SEM if and only if it satisfies the *ordered pair condition* for all ordered pairs of variables (Eberhardt et al., 2010; Hyttinen et al., 2010, 2012a,b). Similar identifiability results can be obtained for (acyclic) causal models with binary variables by assuming a noisy-OR parameterization for the local conditional probability distributions (Hyttinen et al., 2011).

Such identifiability results immediately give rise to the following questions of optimal experiment selection:

- What is the least number of experiments that satisfy the above conditions?
- Can we give procedures to construct such sets of experiments?

The above questions can be raised similarly given additional context, such as the following:

- The number of variables that can be subject to an intervention simultaneously is limited in some way.
- Background knowledge about the underlying causal structure is available.

Naturally, there are other possible scenarios, but we focus on these, since we are aware of their counterparts in the combinatorics literature. To avoid having to repeatedly state the relevant search space assumptions, we will present the remainder of this article in terms of the satisfaction of the (unordered and ordered) pair conditions, which provide the basis for the identifiability results just cited.

3. Correspondence to Separating Systems and Cut-coverings

The satisfaction of the pair conditions introduced in the previous section is closely related to two problems in combinatorics: Finding *(completely) separating systems*, and finding *(directed) cut-coverings*. Throughout, to simplify notation and emphasize the connections, we will overload symbols to the extent that there is a correspondence to the problem of experiment selection for causal discovery.

Definition 4 (Separating System) *A* separating system $\mathcal{C} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_K\}$ *is a set of subsets of an n-set* $\mathcal{V}$ *with the property that given any two distinct elements* $x_i, x_j \in \mathcal{V}$*, there exists a* $\mathcal{I}_k \in \mathcal{C}$ *such that* $x_i \in \mathcal{I}_k \wedge x_j \notin \mathcal{I}_k$ or $x_i \notin \mathcal{I}_k \wedge x_j \in \mathcal{I}_k$.

Definition 5 (Completely Separating System) *A* completely separating system $\mathcal{C} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_K\}$ *is a set of subsets of an n-set* $\mathcal{V}$ *with the property that given any two distinct elements* $x_i, x_j \in \mathcal{V}$*, there exist* $\mathcal{I}_k, \mathcal{I}_{k'} \in \mathcal{C}$ *such that* $x_i \in \mathcal{I}_k \wedge x_j \notin \mathcal{I}_k$ and $x_i \notin \mathcal{I}_{k'} \wedge x_j \in \mathcal{I}_{k'}$.

As can be easily verified, a set of experiments $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ that satisfies the *unordered* pair condition for all pairs over n variables in $\mathcal{V}$ directly corresponds to a separating system over the variable

5. For simplicity, we focus in this article on identifying the causal structure among the observed variables, even though some of the cited results also permit the identification of confounding.

set, while a set of experiments that satisfies the *ordered* pair condition for all ordered variable pairs corresponds to a *completely* separating system over the variables.[6]

A related but more general problem is that of finding cut-coverings. First, we need to define cuts and directed cuts: A *cut* $\mathcal{E}_k$ corresponds to a partition of a set of vertices $\mathcal{V}$ of an undirected graph $H = (\mathcal{V}, \mathcal{P})$ into two sets $\mathcal{I}$ and $\mathcal{U}$. Any edge $p \in \mathcal{P}$ connecting an $x_i \in \mathcal{I}$ to an $x_u \in \mathcal{U}$ is said to be *in the cut* $\mathcal{E}_k$. For a directed graph F, a *directed cut* is a cut where only the edges *from* vertices in $\mathcal{I}$ *to* vertices in $\mathcal{U}$ are in the cut, while edges in the opposite direction are not. We are thus ready to define a *cut-covering* and a *directed cut-covering*:

Definition 6 (Cut-covering) *A cut-covering for an undirected graph $H = (\mathcal{V}, \mathcal{P})$ is a set of cuts $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ such that each edge $p \in \mathcal{P}$ of H is in some cut $\mathcal{E}_k$.*

Definition 7 (Directed Cut-covering) *A directed cut-covering for a directed graph $F = (\mathcal{V}, Q)$ is a set of directed cuts $\{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$ such that each directed edge $q \in Q$ of F is in some directed cut $\mathcal{E}_k$.*

The correspondence of finding cut-coverings to the problem of experiment selection is now immediate: In the case of searching for a set of experiments that satisfies the ordered pair condition for all ordered pairs of variables in $\mathcal{V}$, let the graph $F = (\mathcal{V}, Q)$ be a *complete* directed graph over the vertex set $\mathcal{V}$ where each ordered pair of variables is connected by a directed edge. Each directed edge represents an ordered pair condition for a pair of vertices (x_i, x_j) that needs to be satisfied by the set of experiments. Finding such a set of experiments is then equivalent to finding a directed cut-covering for F, where each experiment corresponds to a directed cut. An analogous correspondence holds for the unordered pair condition with a complete undirected graph H. We discuss the generalization and interpretation of the problem when H or F are not complete graphs in Section 6.

As our overloading of symbols suggests, most aspects of the experiment selection have direct counterparts in the cut-covering representation. However, the edges representing direct causes in a causal graph G *do not* correspond to the edges representing the satisfaction of an ordered pair condition in the directed graph F. That is, F and G in general do not share the same edge structure: G is the graph of the underlying causal model to be identified, while F represents the set of ordered pairs that are not yet satisfied. Moreover, there is a difference in how the causal graph G is changed in light of an experiment $\mathcal{E} = (\mathcal{I}, \mathcal{U})$, and how the ordered pair graph F is changed in light of the (corresponding) cut $\mathcal{E}$. The experiment $\mathcal{E}$ results in the manipulated causal graph G', which is the same as the original causal graph G except that the edges *into* the variables in $\mathcal{I}$ are removed. The corresponding *cut* $\mathcal{E}$, however, cuts the edges of the ordered pair condition graph F that are *outgoing* from variables in $\mathcal{I}$ (and simultaneously *into* variables in $\mathcal{U}$). This may seem unintuitive, but in fact these two representations of $\mathcal{E}$ (the experiment and the cut) illustrate two aspects of what an experiment achieves: It manipulates the underlying causal graph G (by breaking incoming edges on variables in $\mathcal{I}$), and it satisfies the ordered pair condition for all ordered pairs (x_j, x_u) with $x_j \in \mathcal{I}$ and $x_u \in \mathcal{U}$ by determining whether x_j has a causal effect on x_u. Similarly in the unordered case, the causal graph G (or its skeleton) must not be confused with the undirected graph H representing the unordered pairs that are not yet satisfied.

6. *Separating systems* and *completely separating systems* are sometimes also referred to by the terms *weakly separating systems* and *strongly separating systems*, respectively.

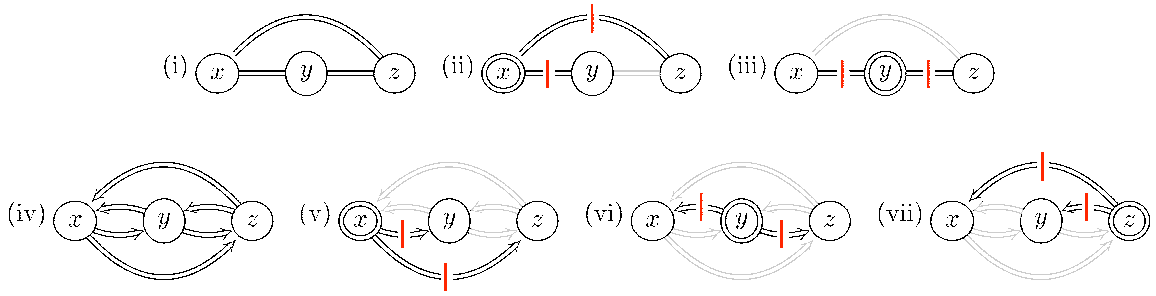


Figure 2: *Top*: Satisfaction of the unordered pair condition for all pairs of variables in a three variable model. Undirected double-lined edges indicate pairs for which the unordered pair condition needs to be satisfied (graph (i)). The cuts (|) indicate which pairs are satisfied by the experiments intervening on x (graph (ii)) and y (graph (iii)). *Bottom*: Satisfication of the ordered pair condition: Directed double-lined edges indicate ordered pairs for which the ordered pair condition needs to be satisfied (graph (iv)). Graphs (v-vii) show which pairs are in the directed cuts (are satisfied) by the respective single-intervention experiments. See text for details.

Figure 2 illustrates for a three variable model (such as that in Figure 1) how the satisfaction of the (un)ordered pair condition for all pairs of variables is guaranteed using cut-coverings. Graph (i) gives the complete *undirected* graph H over $\{x, y, z\}$ illustrating the three unordered pairs for which the *unordered* pair condition needs to be satisfied. Graphs (ii) and (iii) show for which pairs the unordered pair condition is satisfied by a single intervention experiment on x (or y, respectively), that is, which pairs are in the cut (|). The pairs that remain unsatisfied by each experiment, respectively, are shown in gray for easier legibility. Together these experiments constitute a cut-covering for H. Similarly, graph (iv) gives the complete *directed* graph F over the three variables, illustrating the six ordered pairs of variables for which the *ordered* pair condition needs to be satisfied. Graphs (v-vii) show for which pairs the ordered pair condition is satisfied by a single intervention experiment on x (or y or z, respectively), that is which pairs are in the directed cut (|), while the others are again shown in gray. As can be seen, in the ordered case all three experiments are needed to provide a directed cut-covering for F.

The correspondence between the problem of finding experiments that satisfy the pair conditions on the one hand and finding separating systems or cut-coverings on the other, allows us to tap into the results in combinatorics to inform the selection of experiments in the causal discovery problem.

4. Minimal Sets of Experiments

We now turn to the concrete problem of constructing sets of experiments which satisfy the pair condition for all pairs of variables, while simultaneously requiring as few experiments as possible. We divide the results concerning the unordered and the ordered pair condition into Sections 4.1 and 4.2, respectively. In what follows, we always start by presenting the explicit construction of the intervention sets, and subsequently give the available bounds on the number of experiments. The constructions are also available as special cases of the algorithms presented in Section 5, and

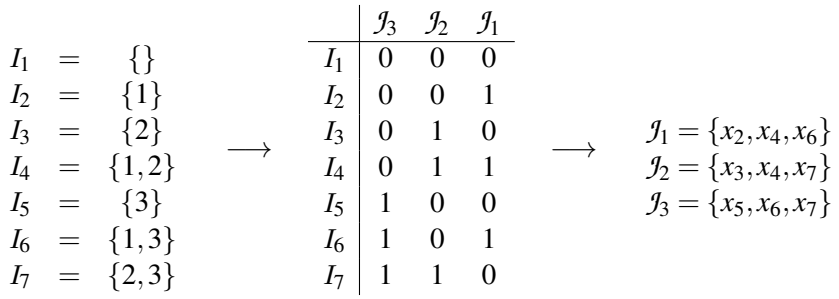


Figure 3: An illustration of the relationship between intervention sets and index sets, giving the construction of a minimal set of experiments satisfying the *unordered* pair condition for all pairs of variables in a 7 variable model. The index sets $I_1, \ldots, I_n$ (left) are chosen as distinct subsets of the set of experiment indexes $\{1,2,3\}$, and each row of the binary matrix (middle) marks the corresponding experiments. The intervention sets $\mathcal{J}_1, \ldots, \mathcal{J}_K$ (right) are then obtained by reading off the columns of this matrix. Note that one additional variable could still be added (intervened on in all three experiments) while still satisfying the unordered pair condition for all variable pairs. For nine variables, however, a minimum of four experiments would be needed.

implemented in our associated code package. Throughout, $i = 1, \ldots, n$ indexes the variables in the variable set $\mathcal{V}$, while $k = 1, \ldots, K$ indexes the experiments in the construction.

Many of the constructions of intervention sets $\mathcal{J}_1, \ldots, \mathcal{J}_K$ will be examined using so-called *index sets* $I_1, \ldots, I_n$, where

$$I_i \quad = \quad \{k \,|\, x_i \in \mathcal{J}_k\}.$$

That is, the i:th index set simply lists the indexes of the intervention sets that include the variable x_i. Clearly, K experiments (or separating sets) over n variables can be defined either in terms of the intervention sets $\mathcal{J}_1, \ldots, \mathcal{J}_K$ or equivalently in terms of the index sets $I_1, \ldots, I_n$. See Figure 3 for an illustration.

4.1 Satisfying the Unordered Pair Condition

The earliest results (that we are aware of) relevant to finding minimal sets of experiments satisfying the unordered pair condition are given in Rényi (1961)[7] in the terminology of separating systems. He found that a separating system $\mathcal{C} = \{\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_K\}$ over $\mathcal{V}$ can be obtained by assigning distinct binary numbers to each variable in $\mathcal{V}$. That is, the strategy is to choose distinct index sets for all variables in $\mathcal{V}$. This is supported by the following Lemma:

Lemma 8 (Index sets must be distinct) *Intervention sets $\{\mathcal{J}_1, \ldots, \mathcal{J}_K\}$ satisfy the unordered pair condition for all unordered pairs of variables if and only if the corresponding index sets $I_1, \ldots, I_n$ are distinct.*

7. Rényi (1961) also examines the probability of finding separating systems when subsets of $\mathcal{V}$ are selected randomly. These results apply to causal discovery when experiments are selected at random.

Proof Assume that the index sets are distinct. Since the index sets I_i and I_j of any two variables are distinct, there must exist an index k such that either $k \in I_i$ and $k \notin I_j$, or $k \notin I_i$ and $k \in I_j$. Thus, the experiment $\mathcal{E}_k$ satisfies the unordered pair condition for the pair $\{x_i, x_j\}$.

Next assume that the unordered pair condition is satisfied for all pairs. Take two arbitrary index sets I_i and I_j. Since the unordered pair condition is satisfied for the pair $\{x_i, x_j\}$, there is an experiment $\mathcal{E}_k$ where x_i is intervened on and x_j is not, or x_j is intervened and x_i is not. Either way, $I_i \neq I_j$. ■

Again, Figure 3 is used to illustrate this concept.

Rényi (1961, p. 76) notes that the smallest separating system for a set of n variables has size

$$c(n) \quad = \quad \lceil \log_2(n) \rceil. \tag{1}$$

This is clear from the previous lemma: K experiments only allow for up to 2^K distinct index sets. Equivalent results and procedures are derived, only in the terminology of finding minimal cut-coverings for complete graphs, by Loulou (1992, p. 303). In the terminology of causal discovery, Eberhardt (2007, Theorem 3.3.4) requires $\lfloor \log_2(n) \rfloor + 1$ experiments to guarantee identifiability of a causal model.[8] For graphs over three variables, the result is obvious given the graph H (for the unordered pair condition) in Figure 2 (top, left): For a cut-covering of the three undirected edges, $2 = \lceil \log_2(3) \rceil$ cuts are necessary and sufficient. The two cuts in graphs (ii) and (iii) in Figure 2 corresponding to the experiments in the last row of Figure 1 are an example. Figure 4 shows the number of experiments needed to satisfy the unordered pair condition for all variable pairs for models of up to 5,000 variables.

4.2 Satisfying the Ordered Pair Condition

When Dickson (1969, p. 192) coined the term "completely separating systems", he also showed that as the number $n = |\mathcal{V}|$ of elements tends to infinity, the size of a minimal completely separating system approaches the size of a (standard) separating system, that is, $\log_2(n)$. However, Dickson did not derive the exact minimal size. Shortly after, Spencer (1970) recognized the connection between completely separating systems and antichains in the subset lattice, as defined below. See Figure 5 for an illustration.

Definition 9 (Antichain) *An antichain (also known as a Sperner system) $\{\mathcal{S}_i\}$ over a set $\mathcal{S}$ is a family of subsets of $\mathcal{S}$ such that $\forall i, j : \mathcal{S}_i \nsubseteq \mathcal{S}_j$ and $\mathcal{S}_i \nsupseteq \mathcal{S}_j$.*

The connection between antichains and completely separating systems (that is, satisfying the ordered pair condition) is then the following:

Lemma 10 (Index sets form an antichain) *The intervention sets $\{\mathcal{I}_1, \ldots, \mathcal{I}_K\}$ satisfy the ordered pair condition for all ordered pairs if and only if the corresponding index sets $\{I_1, \ldots, I_n\}$ form an antichain over $\{1, \ldots, K\}$.*

8. The one additional experiment sometimes required in this case derives from the need to satisfy the ordered pair condition, or unordered pair condition *and* the covariance condition, for each pair of variables, as discussed in Section 2. The covariance condition can be trivially satisfied with a single passive observational data set (a so-called *null*-experiment).

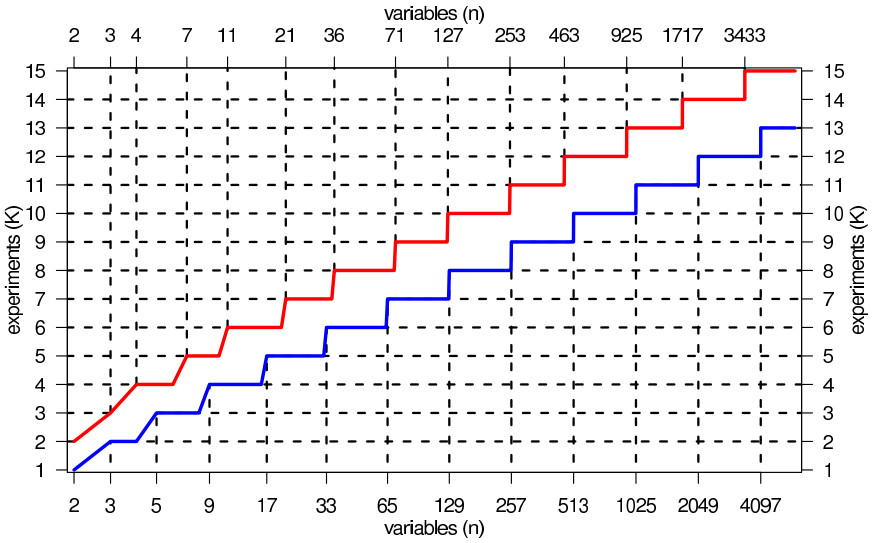


Figure 4: Sufficient and necessary number of experiments needed to satisfy the unordered (blue, lower solid line) and the ordered (red, upper solid line) pair condition for models of different sizes (in log-scale). The number of variables in the models are only ticked on the x-axis when an additional experiment is needed. For example, for a model with 100 variables, 7 experiments are needed to satisfy the unordered pair condition, while 9 experiments are needed to satisfy the ordered pair condition.

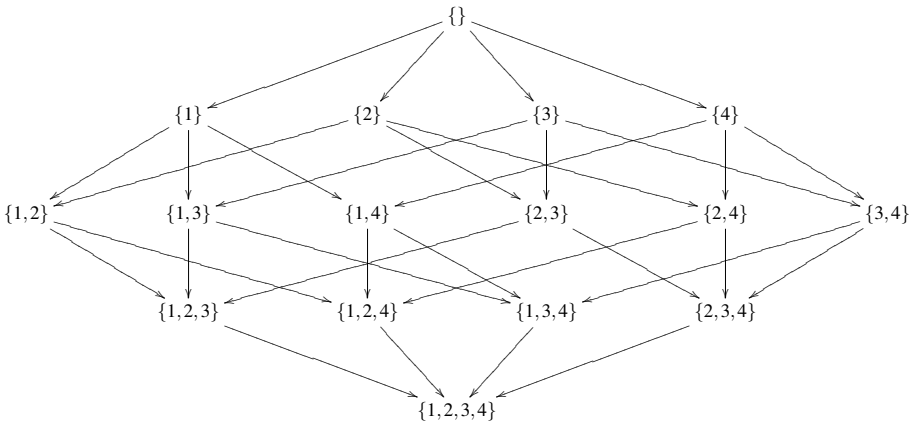


Figure 5: Subset lattice of subsets of $\{1,2,3,4\}$. A directed path from set $\mathcal{S}_i$ to set $\mathcal{S}_j$ exists if and only if $\mathcal{S}_i \subset \mathcal{S}_j$. The largest antichain is the family of sets that are not connected by any directed paths, which in this case is formed by all the sets of size 2, that is, all the sets on the middle 'row' of the graph.

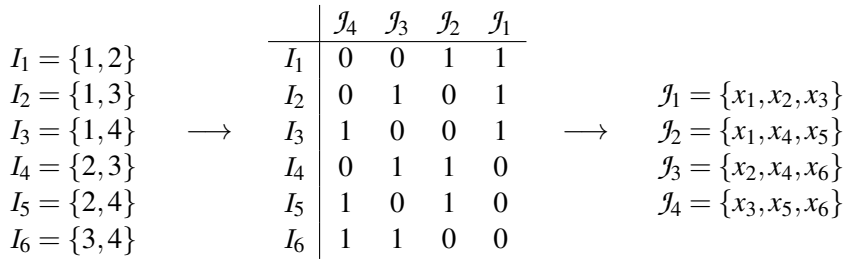


Figure 6: Designing the intervention sets of experiments satisfying the *ordered* pair condition for all ordered pairs of variables in a $n = 6$ variable model with $K = 4$ experiments. Select the index sets $I_1, \ldots, I_n$ as an antichain over $\{1, \ldots, K\}$ and translate the index sets into intervention sets $\mathcal{I}_1, \ldots, \mathcal{I}_K$.

Proof Assume that the index sets form an antichain. Consider an arbitrary ordered pair (x_i, x_j). Since index sets form an antichain we have that $I_i \not\subseteq I_j$ and there must be experiment $\mathcal{E}_k$ such that $k \in I_i$ and $k \notin I_j$. This experiment satisfies the ordered pair condition for the ordered pair (x_i, x_j).

Next assume that the ordered pair condition is satisfied for all ordered pairs. Take two arbitrary index sets I_i and I_j. Since the ordered pair condition is satisfied for the pair (x_i, x_j), there is an experiment where x_i is intervened on and x_j is not, thus $I_i \not\subseteq I_j$. Symmetrically, $I_j \not\subseteq I_i$. Thus the index sets form an antichain. ■

Earlier, Sperner (1928) had already proven the following theorem on the maximum possible size of an antichain.

Theorem 11 (Sperner's Theorem) *The largest antichain over* $\{1, \ldots, K\}$ *is formed by the subsets of constant size* $\lfloor K/2 \rfloor$ *and thus has size* $\binom{K}{\lfloor K/2 \rfloor}$.

Thus, the minimal completely separating system over a set of size n can always be constructed by selecting the corresponding index sets as any distinct $\lfloor K/2 \rfloor$-size subsets. See Figure 6 for an illustration. Using this rationale, Spencer (1970) notes that the cardinality $c(n)$ of a minimal completely separating system for n elements is given by

$$c(n) \quad = \quad \min\{K : \binom{K}{\lfloor K/2 \rfloor} \geq n\}, \tag{2}$$

which can be approximated using Stirling's approximation as

$$c(n) = \log_2(n) + \frac{1}{2}\log_2\log_2(n) + \frac{1}{2}\log_2(\frac{\pi}{2}) + o(1).$$

Equation 2 is re-proven for directed cut-coverings over complete graphs by Alon et al. (2007, Theorem 11), also using the connection to antichains. To our knowledge, tight bounds or constructions have not been previously described in the causal discovery literature on experiment selection.

For graphs over three variables, the graph F (for the ordered pair condition) in Figure 2 (bottom, left) illustrates the point: For a directed cut-covering of the six directed edges, $c(3) = 3$ directed cuts are necessary and sufficient. The three cuts shown in graphs (v-vii) of Figure 2 corresponding to the three possible single-intervention experiments are an example, but the cuts corresponding to the

double-intervention experiments in graphs (ii-iv) of Figure 1 would also work. Figure 4 shows the number of experiments required for models with up to $5,000$ variables. Note that the difference between the number of experiments needed for a separating and a completely separating system over a given number of variables is only 2 or 3 experiments. Thus, in many cases the possibility of applying an inference procedure based on weaker assumptions may be worth the investigative cost of a few additional experiments to satisfy the ordered pair condition.

5. Limiting Intervention Set Size

Section 4 focused on characterizing minimal sets of experiments that guarantee identifiability, but paid no attention to the particular nature of those experiments. In some cases, the experiments might require a simultaneous intervention on half of the variables, but of course such experiments will in many scientific contexts not be feasible. In this section we consider a generalization of the problem of finding the optimal set of experiments that can take into account additional constraints on the size of the intervention sets. We consider the following variants of the problem:

1. Given n variables and K experiments, find intervention sets $\mathcal{I}_1, \ldots, \mathcal{I}_K \subseteq \mathcal{V}$ satisfying the ordered or unordered pair condition for all variable pairs, such that the intervention sets have minimal

   (a) **average** intervention set size $\text{mean}_{k=1}^{K} |\mathcal{I}_k| = \frac{1}{K} \sum_{k=1}^{K} |\mathcal{I}_k|$ (which is equivalent to minimizing the total number of interventions), or

   (b) **maximum** intervention set size $\max_{k=1}^{K} |\mathcal{I}_k|$.

2. Given n variables and a maximum allowed intervention set size r, find the minimum number of experiments $m(n,r)$ for which there exists intervention sets $\mathcal{I}_1, \ldots, \mathcal{I}_{m(n,r)} \subseteq \mathcal{V}$ that satisfy the ordered or unordered pair condition for all variable pairs.

As will become clear from the following discussion, these problems are related. Note that, depending on the additional constraints, these problems may not have solutions (for example, for the unordered case of Problem 1(a) and 1(b), when K is smaller than the bound given in Equation 1), or they may trivially reduce to the problems of the previous sections because the additional constraints are irrelevant (for example, Problem 2 reduces for the unordered case to the problem discussed in Section 4.1 if $r \geq n/2$). As in the previous section, we separate the discussion of the results into those pertaining to the *unordered* pair condition (Section 5.1) and those pertaining to the *ordered* pair condition (Section 5.2). The algorithms presented here can also be used to construct intervention sets that satisfy the bounds discussed in Section 4.

5.1 Limiting the Intervention Set Size for the Unordered Pair Condition

We start with the simplest problem, Problem 1(a) for the *unordered* case, and give the construction of a set of experiments that achieves the smallest possible *average* intervention set size, given the number of variables n and the number of experiments K. The construction we present here is closely related to the first procedures we are aware of, given by Katona (1966). Finding a design which minimizes the average intervention set size is straightforward once one considers the problem in terms of the index sets. The sum of intervention set sizes is of course equal to the sum of index

$$
\begin{array}{lcl}
I_1 & = & \{\} \\
I_2 & = & \{1\} \\
I_3 & = & \{2\} \\
I_4 & = & \{3\} \\
I_5 & = & \{4\} \\
I_6 & = & \{1,2\} \\
I_7 & = & \{3,4\}
\end{array}
\quad \longrightarrow \quad
\begin{array}{c|cccc}
 & \mathcal{J}_4 & \mathcal{J}_3 & \mathcal{J}_2 & \mathcal{J}_1 \\
\hline
I_1 & 0 & 0 & 0 & 0 \\
I_2 & 0 & 0 & 0 & 1 \\
I_3 & 0 & 0 & 1 & 0 \\
I_4 & 0 & 1 & 0 & 0 \\
I_5 & 1 & 0 & 0 & 0 \\
I_6 & 0 & 0 & 1 & 1 \\
I_7 & 1 & 1 & 0 & 0
\end{array}
\quad \longrightarrow \quad
\begin{array}{l}
\mathcal{J}_1 = \{x_2, x_6\} \\
\mathcal{J}_2 = \{x_3, x_6\} \\
\mathcal{J}_3 = \{x_4, x_7\} \\
\mathcal{J}_4 = \{x_5, x_7\}
\end{array}
$$

Figure 7: Intervention sets for experiments that satisfy the *unordered* pair condition for all pairs of variables in a 7 variable model such that both the maximum and average intervention set size are minimized. The index sets were chosen using Algorithm 2.

set sizes (because both represent the total number of interventions):

$$\sum_{k=1}^{K} |\mathcal{J}_k| \;=\; \sum_{i=1}^{n} |I_i|. \tag{3}$$

This identity, together with Lemma 8, implies that to obtain intervention sets with minimum average size, it is sufficient to find the n smallest distinct subsets of $\{1,\ldots,K\}$ as the index sets. There are a total of $\sum_{j=0}^{p}\binom{K}{j}$ index sets I_i with $|I_i| \leq p$. Consequently, the size l of the largest required index set is the integer solution to the inequalities

$$\sum_{j=0}^{l-1}\binom{K}{j} \;<\; n \;\leq\; \sum_{j=0}^{l}\binom{K}{j}. \tag{4}$$

If there is no solution for l, the unordered pair condition cannot be satisfied with K experiments. If there is a solution, the inequalities in Equation 4 imply that when choosing the smallest index sets, we have to select *all*

$$t \;=\; \sum_{j=0}^{l-1}\binom{K}{j}$$

index sets of sizes 0 to $l-1$, and the remaining $n-t$ sets of size l. Since the sum of the intervention set sizes is the same as the sum of the index set sizes (Equation 3), the average intervention set size obtained is

$$\text{mean}_{k=1}^{K}|\mathcal{J}_k| \;=\; \frac{1}{K}\sum_{k=1}^{K}|\mathcal{J}_k| \;=\; \frac{1}{K}\sum_{i=1}^{n}|I_i| \;=\; \frac{1}{K}\left[\sum_{j=0}^{l-1} j\binom{K}{j} + l(n-t)\right]. \tag{5}$$

For K experiments this is the minimum average intervention set size possible that satisfies the unordered pair condition for all pairs among n variables. For the case of $n = 7$ variables and $K = 4$ experiments, Figure 7 provides an example of the construction of intervention sets with minimal average size. Note that the minimum average would not have been affected if the index sets I_6 and I_7 had been chosen differently (but with the same size) as long as all the index sets remained distinct.

Algorithm 1 Selects p index sets of size l, for K experiments, such that the indexes are distributed *fairly* among the index sets. The idea of this algorithm appears in a proof in Cameron (1994, accredited to D. Billington).

Fair(K,l,p)

Draw the index sets $\{I_1,\ldots,I_p\}$ as distinct l-size subsets of $\{1,\ldots,K\}$.

While TRUE,

Find the most frequent index M and the least frequent index m among the index sets $\{I_1,\ldots,I_p\}$.

If $\text{freq}(M)-\text{freq}(m)\leq 1$ then exit the loop.

Find[10] a set $\mathcal{A}$ of size $l-1$ such that $(\{M\}\cup\mathcal{A})\in\{I_1,\ldots,I_p\}$ and $(\{m\}\cup\mathcal{A})\notin\{I_1,\ldots,I_p\}$.

Replace the index set $(\{M\}\cup\mathcal{A})$ with $(\{m\}\cup\mathcal{A})$ in $\{I_1,\ldots,I_p\}$.

Return the index sets $\{I_1,\ldots,I_p\}$.

The minimum average intervention set size in Equation 5 also gives a lower bound for the lowest possible *maximum* intervention set size, given the number of experiments K and variables n (Problem 1(b)): At least one intervention set of size $\lceil \text{mean}_{k=1}^{K}|\mathcal{I}_k|\rceil$ or larger is needed, because otherwise the intervention sets would yield a lower average. Next, we will show that this is also an upper bound.

The size of an arbitrary intervention set $\mathcal{I}_k$ is equal to the number of index sets that contain the index k. We say that index sets are selected *fairly* when the corresponding intervention sets satisfy

$$|\mathcal{I}_k|-|\mathcal{I}_{k'}| \quad\leq\quad 1 \quad \forall k,k'. \tag{6}$$

In the construction that minimizes the average intervention set size, the index sets $I_1,\ldots,I_t$ constitute *all* possible subsets of $\{1,\ldots,K\}$ of size $l-1$ or less, and consequently all indexes appear equally often in these sets.[9] The remaining $n-t$ index sets can be chosen *fairly* using Algorithm 1. It finds fair index sets by simply switching the sets until the experiment indexes appear fairly. Since (i) the average intervention set size remains unchanged by this switching, (ii) the minimum average constitutes a lower bound, and (iii) the intervention set sizes differ by at most one, it follows (see Appendix A) that the lowest maximum intervention set size is given by

$$\max_{k=1}^{K}|\mathcal{I}_k| \quad=\quad \lceil \text{mean}_{k=1}^{K}|\mathcal{I}_k|\rceil. \tag{7}$$

Thus, if the construction of index sets is *fair*, then *both* the minimum average and the smallest maximum intervention set size is achieved, simultaneously solving both Problem 1(a) and 1(b). Algorithm 2 provides this complete procedure. Note that in general it is possible to select index sets such that the maximum intervention set size is not minimized, even though the average is minimal (for example, if I_7 had been $\{1,4\}$ in Figure 7). Figure 8 (top) shows the lowest possible average intervention set sizes. Rounding these figures up to the closest integer gives the lowest possible

9. This is easily seen if the index sets are represented as binary numbers, as in the center of Figure 7. It is also clear from considerations of symmetry.

10. We can always find such a set $\mathcal{A}$: If there were no such set $\mathcal{A}$, then for all sets $\mathcal{B}$ of size $l-1$ such that $(\{M\}\cup\mathcal{B})\in\{I_1,\ldots,I_p\}$ we would also have that $(\{m\}\cup\mathcal{B})\in\{I_1,\ldots,I_p\}$. But then, $\text{freq}(M)-\text{freq}(m)\leq 0$ and the algorithm would have exited already on the previous line.

Algorithm 2 Constructs K intervention sets that satisfy the *unordered* pair condition for all pairs among n variables with a minimum average and smallest maximum intervention set size.

FairUnordered(n, K)

Determine the maximum index set size l from Equation 4, if no such l exists, then the unordered pair condition cannot be satisfied for n variables with K experiments.

Assign all subsets $\mathcal{S} \subseteq \{1,\ldots,K\}$ such that $|\mathcal{S}| \leq l-1$ to index sets $I_1,\ldots,I_t$.

Draw the remaining l-size index sets with: $I_{t+1},\ldots,I_n \leftarrow$ Fair(K, l, $n-t$)

Return the intervention sets $\mathcal{J}_1,\ldots,\mathcal{J}_K$ corresponding to the index sets $I_1,\ldots,I_n$.

Algorithm 3 Constructs K intervention sets that satisfy the *ordered* pair condition for all ordered pairs among n variables and approximates (and sometimes achieves) the minimum average and smallest maximum intervention size.

FairOrdered(n, K)

Determine the maximum index set size l from Equation 10, if no such l exists, then the ordered pair condition cannot be satisfied for n variables with K experiments.

Draw the l-size index sets with: $I_1,\ldots,I_n \leftarrow$ Fair(K, l, n)

Return the intervention sets $\mathcal{J}_1,\ldots,\mathcal{J}_K$ corresponding to the index sets $I_1,\ldots,I_n$.

maximum intervention set sizes. All of these numbers are achieved by intervention sets constructed using Algorithm 2.

Using constructions similar to the above, Katona (1966) and Wegener (1979) were able to derive the following bounds for Problem 2, the minimum number of experiments $m(n,r)$ given an upper bound r on the intervention set sizes:

$$m(n,r) = \lceil \log_2 n \rceil, \quad \text{if } r > n/2, \tag{8}$$

$$\frac{\log_2 n}{\log_2(e \cdot n/r)} \frac{n}{r} \leq m(n,r) \leq \left\lceil \frac{\log_2 n}{\log_2 \lceil n/r \rceil} \right\rceil (\lceil n/r \rceil - 1), \quad \text{if } r \leq n/2, \tag{9}$$

where e denotes Euler's number. Equation 8 just restates Equation 1, since the constructions in Section 4.1 without a constraint on the maximum intervention set size result in intervention sets with no more than $n/2$ variables. For any practical values of n and r, the value of $m(n,r)$ can be found by simply evaluating Equations 4, 5 and 7 for different values of K (starting from the lower bound in 9), so as to find the smallest value of K for which the maximum intervention set size is smaller than or equal to r. Figure 8 (bottom) illustrates the behavior of the function $m(n,r)$ and the bounds given by Equation 9.

5.2 Limiting the Intervention Set Size for the Ordered Pair Condition

Recall from Lemma 10 that satisfaction of the *ordered* pair condition requires that the *n index sets* of a set of K experiments form an antichain over $\{1,\ldots,K\}$. Thus, no matter whether we seek to minimize the average or the maximum intervention set size, we have to ensure that the index sets form an antichain. We begin by considering the (now *ordered* versions of) Problems 1(a) and

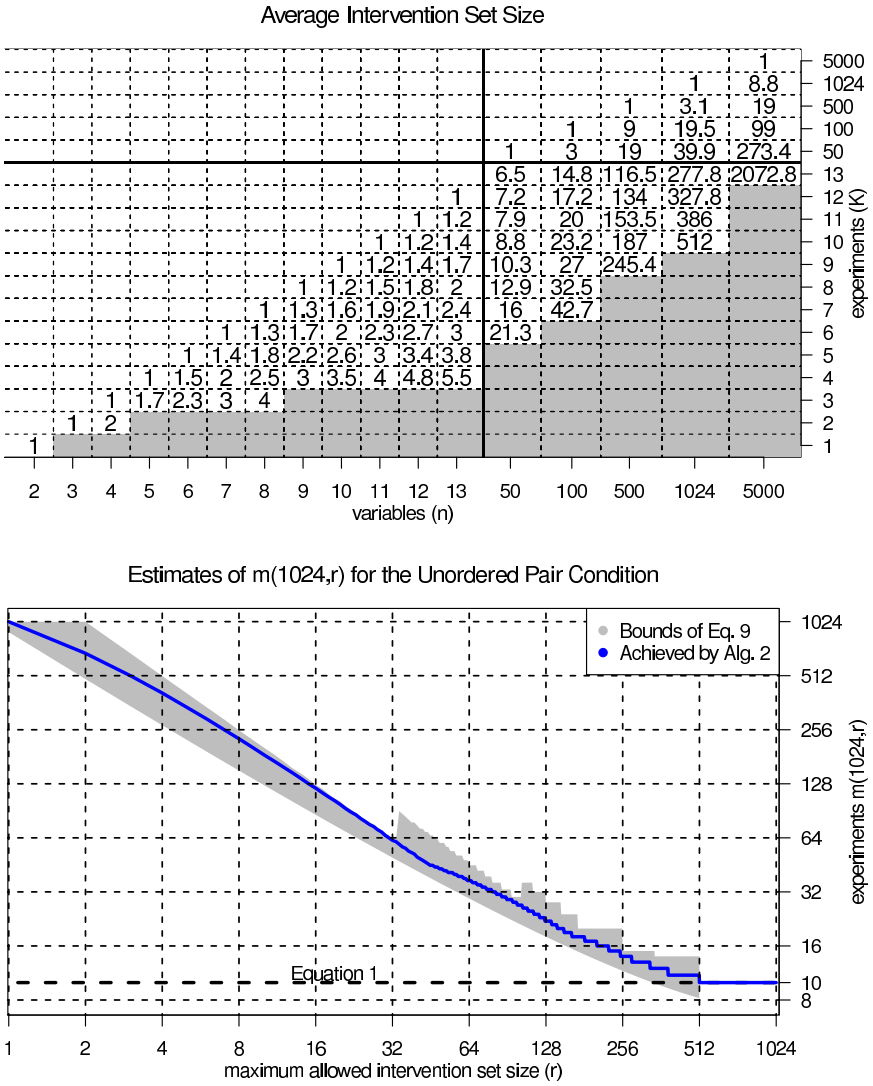


Figure 8: Satisfying the unordered pair condition while limiting the intervention set sizes. *Top:* Lowest achievable average intervention set sizes for models with n variables using K experiments. The lowest achievable maximum intervention set size is the ceiling of the average intervention set size shown in the figure. Grey areas denote an insufficient number of experiments to satisfy the unordered pair condition. Blank areas are uninteresting, since the average intervention set size can be lowered here by including irrelevant passive observational (*null*-)experiments. *Bottom:* The number of experiments needed for $n = 1024$ variables, with a limit r on the maximum allowed intervention set size.

1(b): Given n and K, we want to specify experiments minimizing either the average or maximum intervention set size.

First, we note that to obtain an antichain with n elements from K experiments, at least one of the index sets must be of cardinality l or larger, where l is chosen to satisfy

$$\binom{K}{l-1} \quad < \quad n \quad \leq \quad \binom{K}{l}. \tag{10}$$

This must be the case because it can be shown using the Lemmas in the proof of Theorem 11 (see p. 546 bottom in Sperner, 1928) that the largest antichain with sets of at most size $l-1$ has size $\binom{K}{l-1}$, which—given how l was constructed in Equation 10—is not enough to accommodate all n index sets. On the other hand, it is equally clear that it *is* possible to obtain an antichain by selecting the n index sets to all have sizes l.

Thus, a simple approach to attempt to minimize the *maximum* intervention set size (that is, solve Problem 1 (b)) is to select the n index sets all with sizes l and use Algorithm 3, exploiting Algorithm 1, to construct a *fair* set of index sets. This construction is not fully optimal in all cases because all sets are chosen with size l while in some cases a smaller maximum intervention set size is achievable by combining index sets of different sizes. It is easily seen that Algorithm 3 will generate sets of experiments that have an average and a maximum intervention set size of

$$\text{mean}_{k=1}^{K}|\mathcal{J}_k| \quad = \quad \frac{1}{K}\sum_{k=1}^{K}|\mathcal{J}_k| = \frac{1}{K}\sum_{i=1}^{n}|I_i| = \frac{n\cdot l}{K}, \tag{11}$$

$$\max_{k=1}^{K}|\mathcal{J}_k| \quad = \quad \left\lceil \text{mean}_{k=1}^{K}|\mathcal{J}_k| \right\rceil = \left\lceil \frac{n\cdot l}{K} \right\rceil. \tag{12}$$

Figure 9 (top) shows the maximum intervention set size in the output of Algorithm 3 for several values of n and K. Given some of the subsequent results it can also be shown that some of these are guaranteed to be optimal.[11] While this scheme for solving the directed case of Problem 1(b) is quite good in practice, we are not aware of any efficient scheme that is always guaranteed to minimize the maximum intervention set size.

Alternatively, one may focus on Problem 1 (a) and thus attempt to minimize the *average* intervention set size. For this problem, there exists an efficient procedure that obtains the optimum. Griggs et al. (2012) have recently provided some results in this direction. Here we apply their findings to the problem of experiment selection. To present the construction we need to start with some results pertaining to antichains.

An antichain is said to be *flat* if for any pair of sets I_i and I_j in the antichain, the cardinalities satisfy

$$|I_i| - |I_j| \leq 1, \quad \forall i, j.$$

Note that *flatness* requires a selection of index sets that are *themselves* close in size, while *fairness* (Equation 6) requires a selection of index sets such that the *intervention sets* are close in size. Using this notion of flatness, Lieby (1994) originally formulated the following theorem as a conjecture:

11. But, for example, when $n = 30$ and $K = 8$, Algorithm 3 results in a maximum intervention set size of $\lceil \frac{30\cdot 3}{8} \rceil = 12$, although for this case Algorithm 4 can be used to construct suitable intervention sets with at most 11 members.

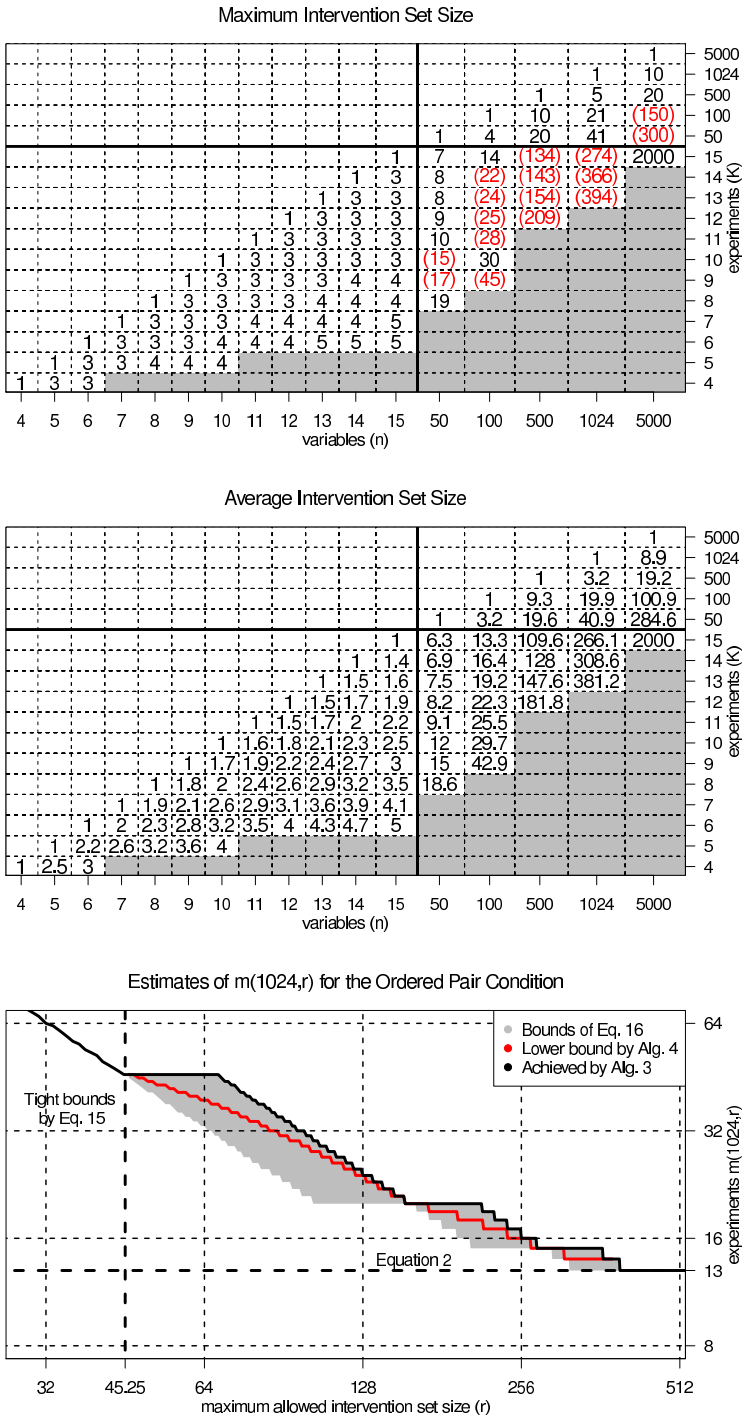


Figure 9: Satisfying the ordered pair condition while limiting the size of the intervention sets. *Top:* Maximum intervention set sizes achieved by Algorithm 3. Black numbers mark the cases where the achieved maximum intervention set size is known to be optimal, while red numbers in parentheses mark cases that are not known to be optimal. *Middle:* Average intervention set sizes achieved by Algorithm 4, all guaranteed to be optimal. *Bottom:* Number of experiments needed to satisfy the ordered pair condition for $n = 1024$ variables with a limit r on the maximum intervention set size.

Theorem 12 (Flat Antichain Theorem) *For every antichain there exists a flat antichain with the same size and the same average set size.*[12]

Since the sum of the index set sizes is identical to the sum of the intervention set sizes, Theorem 12 shows that whenever the ordered pair condition can be satisfied, the average intervention set size can be minimized by a set of flat index sets. From Equation 10 it is thus clear that an antichain minimizing the average set size can be selected solely from the sets of sizes $l-1$ and l. The question then becomes, how can we choose as many sets as possible of size $l-1$, thus minimizing the number of sets needed of size l, nevertheless obtaining a valid antichain?

From the Kruskal-Katona Theorem (Kruskal, 1963; Katona, 1968) it follows that an optimal solution can be obtained by choosing the *first* p index sets of size l and the *last* $n-p$ index sets of size $l-1$ from the *colexicographical order* of each of these sets (separately), defined below.

Definition 13 (Colexicographical Order) *The* colexicographical order *over two sets A and B, where* $|A| = |B|$*, is defined by* $A < B$ *if and only if* $\exists i : (A[i] < B[i]$ *and* $\forall j > i : A[j] = B[j])$*, where* $A[i]$ *denotes the i:th element of the set A, when the elements of the set are arranged in numerical order.*

For example, comparing the sets $A = \{2,3,6\}$ and $B = \{1,4,6\}$ (note that they are already written in numerical order), we obtain $A < B$ because $A[2] = 3$ which is less than $B[2] = 4$, while $A[3] = B[3] = 6$. (See Figure 10 for a further illustration.)

Furthermore, the theory also allows for easily computing the smallest p (and hence largest $n-p$) for which a valid antichain is obtained: Any choice for p can be written in a unique l-cascade form

$$p = \sum_{j=1}^{l} \binom{a_j}{j}, \tag{13}$$

where the integers $a_1, \ldots, a_l$ can be computed using a simple greedy approach (for details see Jukna (2011, p. 146-8) and the code package accompanying this paper). Then, as discussed by Jukna, the number q of sets of size $l-1$ that are not subsets of the first p sets in the appropriate colexicographical order is given by

$$q = \binom{K}{l-1} - \sum_{j=1}^{l} \binom{a_j}{j-1}. \tag{14}$$

Thus, if one can pick the smallest p such that $p+q \geq n$, then it is possible to construct a flat antichain of size n that maximizes the number of index sets with size $l-1$.

Algorithm 4 thus considers all values of p starting from 1, until Equations 13 and 14 imply that there is a flat antichain of size at least n. It then selects the *first* p index sets in the colexicographical order of sets of size l, and the *last* $n-p$ sets in the colexicographical order of sets of size $l-1$. The Kruskal-Katona Theorem ensures that the chosen $(l-1)$-sized sets will not be subsets of the chosen l-sized sets, thereby guaranteeing the antichain property (Jukna, 2011, p. 146-8). See Figure 10 for an example. Thus, Algorithm 4 returns a set of index sets that minimize the average intervention set size, solving (the directed version of) Problem 1 (a). Figure 9 (middle) shows the optimal average sizes for various values of n and K.

12. Partial proofs follow from the work of Kleitman and Milner (1973), Lieby (1999) and Roberts (1999), and quite recently, Kisvölcsey (2006) was able to provide the full proof.

$$
\begin{array}{ccccccccccc}
\{\mathbf{1},\mathbf{2},\mathbf{3}\} & < & \{\mathbf{1},\mathbf{2},\mathbf{4}\} & < & \{\mathbf{1},\mathbf{3},\mathbf{4}\} & < & \{\mathbf{2},\mathbf{3},\mathbf{4}\} & < & \{\mathbf{1},\mathbf{2},\mathbf{5}\} & < & \{\mathbf{1},\mathbf{3},\mathbf{5}\} \\
 & & & & & & & & & & \wedge \\
\{\mathbf{1},\mathbf{3},\mathbf{6}\} & > & \{\mathbf{1},\mathbf{2},\mathbf{6}\} & > & \{\mathbf{3},\mathbf{4},\mathbf{5}\} & > & \{\mathbf{2},\mathbf{4},\mathbf{5}\} & > & \{\mathbf{1},\mathbf{4},\mathbf{5}\} & > & \{\mathbf{2},\mathbf{3},\mathbf{5}\} \\
\wedge & & & & & & & & & & \\
\{\mathbf{2},\mathbf{3},\mathbf{6}\} & < & \{\mathbf{1},\mathbf{4},\mathbf{6}\} & < & \{\mathbf{2},\mathbf{4},\mathbf{6}\} & < & \boxed{\{\mathbf{3},\mathbf{4},\mathbf{6}\}} & < & \{1,5,6\} & < & \cdots
\end{array}
$$

$$
\begin{array}{ccccccccccc}
\cdots & < & \{3,6\} & < & \{4,6\} & < & \boxed{\{\mathbf{5},\mathbf{6}\}} & < & \{\mathbf{1},\mathbf{7}\} & < & \{\mathbf{2},\mathbf{7}\} \\
 & & & & & & & & & & \wedge \\
\{\mathbf{2},\mathbf{8}\} & > & \{\mathbf{1},\mathbf{8}\} & > & \{\mathbf{6},\mathbf{7}\} & > & \{\mathbf{5},\mathbf{7}\} & > & \{\mathbf{4},\mathbf{7}\} & > & \{\mathbf{3},\mathbf{7}\} \\
\wedge & & & & & & & & & & \\
\{\mathbf{3},\mathbf{8}\} & < & \{\mathbf{4},\mathbf{8}\} & < & \{\mathbf{5},\mathbf{8}\} & < & \{\mathbf{6},\mathbf{8}\} & < & \{\mathbf{7},\mathbf{8}\} & &
\end{array}
$$

Figure 10: Selecting the index sets in colexicographical order for $n = 30$ and $K = 8$. Selecting 16 index sets of size 3 (up to $\{3,4,6\}$, in bold) and 14 index sets of size 2 (starting from $\{5,6\}$, in bold), gives a total of 30 index sets and achieves the lowest possible average intervention set size for the given n and K. Note that none of the selected index sets is a subset of another, thus the sets form an antichain. If we were to select only 15 index set of size 3 (up to $\{2,4,6\}$), we could still only select 14 index sets of size 2 (from $\{5,6\}$), ending up with only 29 index sets. If we were to select 17 index sets of size 3 (up to $\{1,5,6\}$), we could select 13 index sets of size 2 (from $\{1,7\}$), and find 30 index sets, but the average intervention set size would then be $1/8$ higher.

Algorithm 4 Obtains a set of K intervention sets satisfying the *ordered* pair condition for all ordered pairs among n variables that minimizes the average intervention set size.

Flat(n, K)

Determine the maximum index set size l from Equation 10, if no such l exists, the ordered pair condition cannot be satisfied for n variables with K experiments.

For p from 1 to n,

Find coefficients $a_1, \ldots, a_l$ for a cascade presentation of p in Equation 13.

Calculate the number q of available index sets of size $l-1$ by Equation 14.

If $p+q \geq n$ exit the for-loop.

Choose the index sets $I_1, \ldots, I_p$ as the *first* sets of size l in the colexicographical order.

Choose the index sets $I_{p+1}, \ldots, I_n$ as the *last* sets of size $l-1$ in the colexicographical order.

Return the intervention sets $\mathcal{J}_1, \ldots, \mathcal{J}_K$ corresponding to the index sets $I_1, \ldots, I_n$.

Trivially, the ceiling of the minimum average intervention set size for n variables in K experiments gives a lower bound on the lowest maximum intervention set size, that is, for Problem 1 (b). This allows us to determine the optimality of some of the outputs of Algorithm 3 in Figure 9 (top).

Problem 2 reverses the free parameters and asks for the minimum number of experiments $m(n,r)$ given a limit r on the maximum size of any intervention set. Cai (1984b) shows that

$$m(n,r) \quad = \quad \left\lceil \frac{2n}{r} \right\rceil, \quad \text{if} \quad 2 \leq \frac{1}{2}r^2 < n. \tag{15}$$

With input $K = \lceil 2n/r \rceil$, Algorithm 3 generates intervention sets of at most size r (see Appendix B)—this verifies that $m(n,r) \leq \lceil 2n/r \rceil$. Cai's result also gives an exact minimum number of experiments when the maximum intervention set size has to be small (see Figure 9 (bottom)). It can also be used to construct a lower bound on the maximum intervention set size when the number of experiments K is given: If $m(n,r) > K$ for some r and K, then the maximum intervention set size given n variables and K experiments must be at least $r+1$. Again, we use this connection to determine the optimality of some of the outputs of Algorithm 3 in Figure 9 (top).

For cases when r does not satisfy the restrictions of Cai's result, Kündgen et al. (2001) use a similar construction to provide the following bounds:[13]

$$\min\{K | n \leq \binom{K}{\lceil Kr/n \rceil}\} \quad \leq \quad m(n,r) \; \leq \; \min\{K | n \leq \binom{K}{\lfloor Kr/n \rfloor}\}, \quad \text{if} \quad r \leq \frac{n}{2}. \tag{16}$$

Again the upper bound can be easily verified: With the upper bound K as input, Algorithm 3 will generate intervention sets of at most size r (see Appendix C). The lower bound is an application of classic results of Kleitman and Milner (1973) concerning average index set sizes. In many cases we can get an improved lower bound on $m(n,r)$ using Algorithm 4 (which optimally minimizes the average number of interventions per experiment, for given n and K): Find the smallest K such that Algorithm 4 returns intervention sets with an average size less than r. In this case we know that the minimum number of experiments given a maximum intervention set size of r must be at least K (see Figure 9 (bottom)).

Finally, note that Ramsay and Roberts (1996) and Ramsay et al. (1998) have considered the problem equivalent to finding a set of experiments where instead of a limited maximum intervention set size, the intervention sets are constrained to have *exactly* some given size. Sometimes more experiments are needed in order to satisfy this harder constraint.

6. Background Knowledge

The results and procedures of the previous two sections apply to scenarios in which there is no background knowledge concerning the possible causal relationships among the variables. In those cases, for complete identification, the chosen experiments must satisfy the (ordered or unordered, depending on the assumptions) pair condition for all pairs of variables. However, in many cases there exists domain knowledge that can assist in inferring the underlying system. For instance, it may be the case that background knowledge rules out certain causal relationships. Hyttinen et al. (2010) gave details of how, in the linear case, such prior knowledge can be integrated into the discovery procedure, reducing the set of ordered pair conditions that need to be satisfied by the experiments. As another example, under causal sufficiency, acyclicity, and faithfulness, if by prior knowledge a given edge is known to be present, or it is known that there is no causal relation between a given pair of variables, this translates directly to an unordered pair that does *not* need to be satisfied

13. Roberts and Rylands (2005) give the exact values for $m(n,r)$ for $n \leq 10$ variables and all suitable r.

by the set of experiments. Finally, any background knowledge that is equivalent to knowledge of the outcome of some experiment can be described in terms of satisfied pair conditions. In this section, we thus consider the selection of experiments when the experiments only need to satisfy the pair condition for a given subset of all variable pairs, but acknowledge that not all background knowledge is representable in terms of satisfied pair conditions.

When the pair condition only needs to be satisfied for a subset of the variable pairs, the search problem is equivalent to that of finding a minimal cut-covering of a given graph. As described in Section 3, we represent the satisfaction of the *unordered* pair condition by a graph H over the vertices $\mathcal{V}$, where an undirected edge between a pair of variables indicates that the unordered pair condition is *not* yet satisfied for that pair (and hence needs to be satisfied by the experiments we select), while the absence of an edge indicates it is already satisfied for the pair (and does not need to be satisfied by our experiments). Similarly, we use a *directed* graph F to represent the satisfaction of the *ordered* pair condition, in the analogous way. Essentially, the combinatorial problems discussed in the two previous sections can thus be interpreted as finding a minimal cut-covering for a *complete* directed or undirected graph, while in this section we consider the problem of finding a minimal cut-covering for an *arbitrary* directed or undirected graph.[14]

First, consider the satisfaction of the unordered pair condition for an arbitrary subset of all variable pairs. Unlike the case without background knowledge, discussed in Sections 4.1, the problem of finding the smallest set of experiments to satisfy the unordered pair condition for a subset of all pairs is known to be hard. Cai (1984a) establishes the connection to minimal graph colorings by showing (in his Theorem 5) that the smallest cardinality $c(H)$ of a cut-covering of an undirected graph H relates to its chromatic number $\chi(H)$ (the smallest number of colors required to vertex-color graph H) as

$$c(H) \quad = \quad \lceil \log_2(\chi(H)) \rceil. \tag{17}$$

The result indicates that the main constraint to reducing the number of experiments are cliques of variables for which the unordered pair condition is not satisfied. Equation 17 constitutes a generalization of the results shown in Section 4.1. Furthermore, it follows from Cai's Theorem 6 that the problem of finding a minimal set of experiments given background knowledge for arbitrary pairs is NP-hard, though constructing the appropriate experiments *given* a graph coloring is very simple. Various approximation algorithms used for graph coloring could be applied, see for example Welsh and Powell (1967), Motwani and Naor (1993), Halldórsson (1993), Bussieck (1994) and Liberti et al. (2011) for proposals, bounds and simulations. Algorithm 5 calls a graph coloring method (in the code package we use the simple approximation algorithm by Welsh and Powell, 1967) and constructs intervention sets based on the graph coloring. It results in $\lceil \log_2(\chi(H)+c) \rceil$ experiments, where c is the number of colors the coloring algorithm uses in excess of the chromatic number $\chi(H)$. So it achieves Cai's optimal bound on the minimum number of experiments (Equation 17) if the coloring method uses the smallest number of colors.

In certain restricted cases the problem is easier. When the underlying model is known to be acyclic and causally sufficient, and the background knowledge derives from passive observational data or suitable previous experiments, the knowledge can be represented in terms of an (interventional) Markov equivalence class. These are sets of causally sufficient acyclic causal models that are

14. Note that Cai (1984a) uses the terminology of 'separating systems' in relation to arbitrary graphs, but this use of the terminology does not seem to be in widespread use so we do not adopt it here.

Algorithm 5 Constructs a set of intervention sets satisfying the *unordered* pair condition for a given *arbitrary* set of unordered pairs represented by an undirected graph H over n variables. The algorithm is adapted from the proof of Theorem 5 in Cai (1984a).

BackgroundUnordered(H)

Obtain a partition $\mathrm{Col}(H)$ of the variables $\mathcal{V}$ of H into q color classes $C_1,\ldots,C_q$ (where $\chi(H) \leq q \leq n$, $\chi(H)$ is the chromatic number of H and $n = |\mathcal{V}|$), such that no adjacent vertices belong to the same class (for example, using the approximation algorithm in Welsh and Powell, 1967).

Let $K = \lceil \log_2(q) \rceil$.

Obtain "intervention sets" $\mathcal{I}'_1,\ldots,\mathcal{I}'_K$ over the color classes $C_1,\ldots,C_q$ that satisfy the unordered pair condition for all pairs of the q color classes (for example, by calling FairUnordered(q,K) (Algorithm 2)).

For each k from 1 to K,

Determine $\mathcal{I}_k = \bigcup_{C_i \in \mathcal{I}'_k} C_i$. Intervention set $\mathcal{I}_k$ consists of all variables colored with a color in $\mathcal{I}'_k$.

Return intervention sets $\mathcal{I}_1,\ldots,\mathcal{I}_K$.

indistinguishable given passive observational data or the experiments performed so far. Algorithm 3 in Hauser and Bühlmann (2012) constructs in *polynomial* time a minimal set of experiments that is sufficient and in the worst case necessary to identify the true causal model within such an (interventional) Markov equivalence class, assuming the skeleton of the true causal graph is identifiable given the set of experiments. We can translate this situation to our framework as follows: With such background knowledge, for complete identifiability, the *un*ordered pair condition only needs to be satisfied for the pairs of variables *adjacent* in the skeleton, for which the orientation of the edge is unknown. Thus, if we again consider the example graph (i) in Figure 11 (repeating graph (i) from Figure 1), and assume that we have a passive observational data set to determine the Markov equivalence class (graph (ii)), then the undirected graph H representing the pairs for which the unordered pair condition remains *un*satisfied is given by graph (iii). (In this case it has the same structure as the skeleton of the Markov equivalence class of the true graph, but that is not generally the case.) Given H (graph (iii)), it is obvious that a single experiment $\mathcal{E} = (\mathcal{J}, \mathcal{U}) = (\{y\}, \{x,z\})$ would resolve the remaining pairs, that is, the single cut $\mathcal{E}$ is a minimal cut-covering (graph (iv)). Under the assumption of acyclicity, causal sufficiency and faithfulness, this is sufficient for identifiability of the causal structure.

Satisfaction of the *ordered* pair condition for an arbitrary subset of the ordered variable pairs is also known to be hard. Cai (1984a) shows in his Theorem 7 that the problem of determining a minimal cut-covering for an arbitrary directed graph is NP-hard. More recently, Watanabe et al. (2000) offered the following bounds on the cardinality of the minimal directed cut-covering $c(F)$:

$$\log_2(\chi(F)) \quad \leq \quad c(F) \quad \leq \quad \lceil \log_2(\chi(F)) \rceil + \lceil \log_2 \lceil \log_2(\chi(F)) + 1 \rceil \rceil,$$

where $\chi(F)$ is the chromatic number of the directed graph F representing the unsatisfied *ordered* pair conditions.[15] These bounds constitute a generalization of the results in Section 4.2 where no background knowledge was assumed. In general, the conversion of background knowledge to

15. Strictly speaking $\chi(F)$ is the chromatic number of the undirected graph with the same set of vertices as F and in which an undirected edge exists between a pair of vertices if an only if they are adjacent in F.

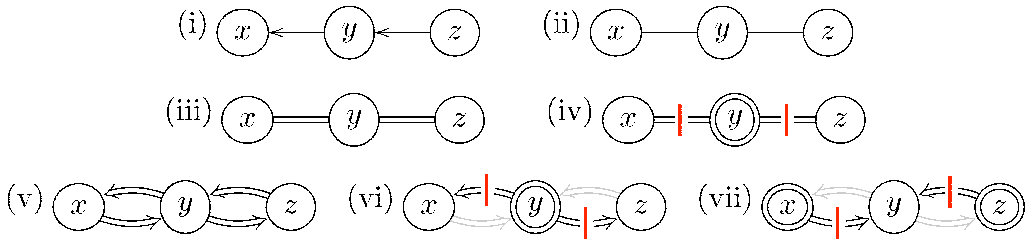


Figure 11: Graph (i) is the true causal generating model (repeating Figure 1, (i)), graph (ii) shows the corresponding passive observational Markov equivalence class (MEC). Graph (iii) illustrates the remaining pairs for which the *unordered* pair condition is not satisfied given the MEC in (ii), and graph (iv) shows that a single intervention on y resolves these pairs, that is, it provides a cut-covering. Given background knowledge obtained from a passive observational data set of graph (i), graph (v) shows the ordered pairs for which the *ordered* pair condition remains unsatisfied. In this case two further experiments are required to provide a directed cut-covering, one intervening on y (graph (vi)) and one intervening on x and z simultaneously (graph (vii)).

pairs satisfying the *ordered* pair condition is more complicated because the equivalence classes of causal structures whose identifiability depends on the satisfaction of the ordered pair condition is less well understood (consider, for example, the equivalence classes for linear cyclic models over a causally insufficient set of variables). But for simple cases it can still be done: Given background knowledge derived from a passive observational data set over graph (i) in Figure 11, graph (v) is the directed graph F indicating the ordered pairs for which the ordered pair condition remains unsatisfied. Graphs (vi) and (vii) then show the two directed cuts that are still required to obtain a directed cut-covering of F. If we assume linearity, but not acyclicity or causal sufficiency, the two experiments corresponding to these cuts would be necessary and sufficient for identifiability given the background knowledge represented in graph (v).

7. Discussion: Related Work and Open Problems

The combinatorial results and procedures we have translated and combined in this paper update and generalize a variety of results in the causal discovery literature on the selection of experiments. For example, Eberhardt (2007, Theorem 3.3.17) only provides an upper bound on the minimum number of experiments sufficient for the satisfaction of the ordered pair condition. With Spencer's result on completely separating systems (Equation 2) we now have an exact result and the experiments that satisfy this result can be constructed using Algorithm 3. Similarly, Eberhardt (2007, Theorem 3.3.29) gave an upper bound on the minimum number of experiments sufficient to satisfy the unordered pair condition when the maximum intervention set size was restricted. The translation of the results of Katona (1966) and Wegener (1979) in Equations 8 and 9 now provide much better bounds, and Algorithm 2 can be used to construct the appropriate intervention sets.

Tong and Koller (2001) and Murphy (2001) use a greedy Bayesian procedure to select the next best single-intervention experiment, but given the computational complexity cannot solve for models with more than five variables. Meganck et al. (2005), He and Geng (2008), Eberhardt (2008) and

Hauser and Bühlmann (2012), in their different ways, also try to find the best next experiment given background knowledge, typically knowledge of the (interventional) Markov equivalence class. In Section 6 we showed that the complexity of the satisfaction of the (un)ordered pair condition given background knowledge is known to be NP-hard in general, and Cai (1984a) already showed (though in the terminology of separating systems) that the minimum number of experiments required is a function of the chromatic number of the graph of unsatisfied pair conditions. Except for special cases (see, for example, Hauser and Bühlmann, 2012), graph-coloring heuristics seem to provide the best approach to problems of this type, and we have implemented one such procedure in Algorithm 5.

In addition, a variety of open problems remain: In Section 5.2 we noted that for the ordered pair condition we are not aware of a general algorithm that generates intervention sets for which the maximum intervention set size is minimized. We also do not know whether the maximum and average intervention set size can be minimized simultaneously (as we showed is possible for the unordered pair condition in Section 5.1). Nevertheless, for both cases we have shown that Algorithm 3 provides a very good approximation and can be shown to provide optimal output in many cases.

More generally, the type of background knowledge we considered in Section 6 may have to be integrated into a search procedure that is subject to constraints on the size of the intervention sets. How to compute the optimal set of experiments in such cases is an open combinatorical problem, for which we are not aware of any solutions that are not brute force searches.

Naturally, there are also further generalizations of the problem settings we considered. For example, it will often not be possible to perform all desired experiments. Some experiments will be more expensive than others or certain combinations of variables may not be manipulable simultaneously. Given a restricted set of experiments, how to select the smallest subset that preserves the discriminatory power of the full set, is known as the "test collection problem". There is a large literature on variants of this problem. Halldórsson et al. (2001) describe the connection to finding minimal cut-coverings and analyze the complexity of the problem. They show that even good approximation algorithms are hard to obtain for this problem, but Moret and Shapiro (1985) analyze and test a variety of heuristics and show that in a real world setting there is reason for optimism that an (almost) minimal set of experiments can still be found relatively easily.

8. Conclusion

We have summarized and presented combinatorial results for the optimal selection of experiments when the goal is to learn the causal structure of a system. Most results were originally derived for so-called *(completely) separating systems* or *minimal cut-coverings*. We used these results to specify the minimum number of experiments necessary and sufficient for identifiability when there is no background knowledge (Section 4), when there are limitations on the size of the intervention sets (Section 5), and when background knowledge is available (Section 6). Where possible, we presented algorithms that actually construct the experiments that satisfy (or closely approximate) the specified bounds and constraints, and we indicated where extant heuristics can be applied. We hope that the constructive way of presenting the results and how to obtain them, may also provide useful guidelines on which experiments to conduct in settings with assumptions not considered here.

For the combinatorics community, we have provided a novel area of application. We have also given a unifying, more easily understandable framework for the set constructions (which are other-

wise often hidden in the proofs of the reported bounds), along with clear examples and computer code. This should help in understanding and comparing the different bounds and constructions. Perhaps this compilation also helps in identifying where new theoretical findings would also have a practical value. We hope that this note provides a translation aid and helps to produce a more congenial flow of research problems and results between the fields of study.

Acknowledgments

The authors would like to thank M. Koivisto and P. Kaski for helpful discussions and three anonymous reviewers for their helpful comments that improved the article. A.H. and P.O.H. were supported by the Academy of Finland. F.E. was supported by a grant from the James S. McDonnell Foundation on 'Experimental Planning and the Unification of Causal Knowledge'.

Appendix A. Proof of Equation 7

Marking the largest intervention set by $\mathcal{J}_h$ we have that

$$\begin{aligned} |\mathcal{J}_h| - \text{mean}_{k=1}^{K} |\mathcal{J}_k| &= \frac{1}{K} \sum_{k=1}^{K} (|\mathcal{J}_h| - |\mathcal{J}_k|) \quad || \text{ Equation 6}, h \in \{1, \ldots, K\} \\ &\leq \frac{K-1}{K} < 1. \end{aligned}$$

Since the maximum intervention set size is an integer which is lower bounded by the average intervention set size, yet less than one above it, Equation 7 follows directly.

Appendix B. Upper Bound of Cai (1984b)

We verify the upper bound in Equation 15: The minimum number of experiments $m(n,r)$, given a limit r on the maximum intervention set size, has an upper bound of $\lceil 2n/r \rceil$, when $n > \frac{1}{2}r^2$. Consider that Algorithm 3 is run with input n and $K = \lceil 2n/r \rceil$. The first step is to find l that satisfies Equation 10. The upper bound in Equation 10 is satisfied when $l = 2$ under the assumption that $n > \frac{1}{2}r^2$:

$$\begin{aligned} \binom{K}{l} &= \frac{K(K-1)}{2} \quad || K \geq \frac{2n}{r} \\ &\geq \frac{2n(K-1)}{2r} \quad || K \geq \frac{2n}{r} > \frac{2\frac{1}{2}r^2}{r} = r \Rightarrow K - 1 \geq r \\ &\geq \frac{2nr}{2r} = n. \end{aligned}$$

Thus, the used index set size l will be at most two, as there exists an antichain of n index sets of constant size $l \leq 2$ over $K = \lceil 2n/r \rceil$ experiments. Then, Algorithm 3 will produce intervention sets

with average size (Equation 11) bounded by r:

$$\begin{aligned} \text{mean}_{k=1}^{K}|\mathcal{I}_k| &= \frac{n \cdot l}{K} \quad ||\frac{1}{K} \leq \frac{r}{2n} \\ &\leq \frac{nlr}{2n} \quad ||l \leq 2 \\ &\leq \frac{2nr}{2n} = r. \end{aligned}$$

Because the index sets are chosen fairly, the maximum intervention set size (Equation 12) is also bounded by integer r:

$$\max_{k=1}^{K}|\mathcal{I}_k| = \lceil \text{mean}_{k=1}^{K}|\mathcal{I}_k| \rceil \leq r.$$

Appendix C. Upper Bound of Kündgen et al. (2001)

We verify the upper bound in Equation 16: The minimum number of experiments $m(n,r)$, given a limit r on the maximum intervention set size, has an upper bound of $\min\{K' | n \leq \binom{K'}{\lfloor K'r/n \rfloor}\}$, when $r \leq n/2$. Consider that Algorithm 3 is run with input n and $K = \min\{K' | n \leq \binom{K'}{\lfloor K'r/n \rfloor}\}$. The first step is to find l that satisfies Equation 10. The upper bound in Equation 10 is satisfied when $l = \lfloor Kr/n \rfloor$, simply by the definition of K. Thus, the used index set size l will be at most $\lfloor Kr/n \rfloor$, as there exists an antichain of n index sets of constant size $l \leq Kr/n$ over K experiments. Then, Algorithm 3 will produce intervention sets with average size (Equation 11) bounded by r:

$$\begin{aligned} \text{mean}_{k=1}^{K}|\mathcal{I}_k| &= \frac{n \cdot l}{K} \quad ||l \leq \frac{Kr}{n} \\ &\leq \frac{nKr}{Kn} = r. \end{aligned}$$

Because the index sets are chosen fairly, the maximum intervention set size (Equation 12) is also bounded by integer r:

$$\max_{k=1}^{K}|\mathcal{I}_k| = \lceil \text{mean}_{k=1}^{K}|\mathcal{I}_k| \rceil \leq r.$$

References

N. Alon, B. Bollobás, A. Gyárfás, J. Lehel, and A. Scott. Maximum directed cuts in acyclic digraphs. *Journal of Graph Theory*, 55(1):1–13, 2007.

E. Bareinboim and J. Pearl. In *Proceedings of the Twenty-Eight Conference on Uncertainty in Artificial Intelligence*, 2012.

M. Bussieck. The minimal cut cover of a graph. Technical Report TR-94-02, Pennsylvania State University, 1994.

M.-C. Cai. On separating systems of graphs. *Discrete Mathematics*, 49(1):15 – 20, 1984a.

M.-C. Cai. On a problem of Katona on minimal completely separating systems with restrictions. *Discrete Mathematics*, 48(1):121 – 123, 1984b.

P. J. Cameron. *Combinatorics: Topics, Techniques, Algorithms*. Cambridge University Press, 1994.

T. Claassen and T. Heskes. Causal discovery in multiple models from different experiments. In *Advances in Neural Information Processing Systems 23*, pages 415–423, 2010.

G. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1999.

T. J. Dickson. On a problem concerning separating systems of a finite set. *Journal of Combinatorial Theory*, 7(3):191 – 196, 1969.

D. Eaton and K. Murphy. Exact Bayesian structure learning from uncertain interventions. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, Journal of Machine Learning Research Workshop and Conference Proceedings 2, 2007.

F. Eberhardt. *Causation and Intervention*. PhD thesis, Carnegie Mellon, 2007.

F. Eberhardt. Almost optimal intervention sets for causal discovery. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2008.

F. Eberhardt, C. Glymour, and R. Scheines. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2005.

F. Eberhardt, P. O. Hoyer, and R. Scheines. Combining experiments to discover linear cyclic models with latent variables. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, Journal of Machine Learning Research Workshop and Conference Proceedings 9, 2010.

R. A. Fisher. *The Design of Experiments*. Hafner, 1935.

J. R. Griggs, Sven Hartman, Uwe Leck, and I. T. Roberts. Full and maximal squashed flat antichains of minimum weight. Technical report, 2012.

B. Halldórsson, M. Halldórsson, and R. Ravi. On the approximability of the minimum test collection problem. In Friedhelm auf der Heide, editor, *Algorithms ESA 2001*, volume 2161 of *Lecture Notes in Computer Science*, pages 158–169. Springer, 2001.

M. Halldórsson. A still better performance guarantee for approximate graph coloring. *Information Processing Letters*, 45(1):19 – 23, 1993.

A. Hauser and P. Bühlmann. Two optimal strategies for active learning of causal models from interventions. In *Proceedings of the 6th European Workshop on Probabilistic Graphical Models*, 2012.

Y.-B. He and Z. Geng. Active learning of causal networks with intervention experiments and optimal designs. *Journal of Machine Learning Research*, 9:2523–2547, 2008.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Causal discovery for linear cyclic models with latent variables. In *Proceedings of the Fifth European Workshop on Probabilistic Graphical Models*, 2010.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Noisy-or models with latent confounding. In *Proceedings of the Twenty-Seventh Conference Conference on Uncertainty in Artificial Intelligence*, 2011.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Causal discovery of linear cyclic models from multiple experimental data sets with overlapping variables. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 2012a.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Learning linear cyclic causal models with latent variables. *Journal of Machine Learning Research*, 13:3387–3439, 2012b.

S. Jukna. *Extremal Combinatorics*. Springer, 2011.

G. Katona. On separating systems of a finite set. *Journal of Combinatorial Theory*, 1(2):174 – 194, 1966.

G. Katona. A theorem of finite sets. In P. Erdos and G. Katona, editors, *Theory of graphs*. Akademiai Kiado and Academic Press, 1968.

Á. Kisvölcsey. Flattening antichains. *Combinatorica*, 26:65–82, 2006.

D.J. Kleitman and E.C. Milner. On the average size of the sets in a Sperner family. *Discrete Mathematics*, 6(2):141 – 147, 1973.

J. B. Kruskal. The number of simplices in a complex. In R. Bellman, editor, *Mathematical Optimization Techniques*. University of California Press, 1963.

A. Kündgen, D. Mubayi, and P. Tetali. Minimal completely separating systems of k-sets. *Journal of Combinatorial Theory, Series A*, 93(1):192 – 198, 2001.

L. Liberti, L. Alfandari, and M.-C. Plateau. Edge cover by connected bipartite subgraphs. *Annals of Operations Research*, 188:307–329, 2011.

P. Lieby. The separation problem. Honours thesis, Northern Territory University, 1994.

P. Lieby. *Extremal Problems in Finite Sets*. PhD thesis, Northern Territory University, 1999.

R. Loulou. Minimal cut cover of a graph with an application to the testing of electronic boards. *Operations Research Letters*, 12(5):301 – 305, 1992.

S. Meganck, B. Manderick, and P. Leray. A decision theoretic approach to learning Bayesian networks. Technical report, Vrije Universiteit Brussels, 2005.

B. M. E. Moret and H. D. Shapiro. On minimizing a set of tests. *SIAM Journal on Scientific and Statistical Computing*, 6(4):983–1003, 1985.

R. Motwani and J. Naor. On exact and approximate cut covers of graphs. Technical report, Stanford University, 1993.

K. P. Murphy. Active learning of causal Bayes net structure. Technical report, U.C. Berkeley, 2001.

J. Pearl. *Causality*. Oxford University Press, 2000.

C. Ramsay and I. T. Roberts. Minimal completely separating systems of sets. *Australasian Journal of Combinatorics*, 13:129–150, 1996.

C. Ramsay, I. T. Roberts, and F. Ruskey. Completely separating systems of k-sets. *Discrete Mathematics*, 183:265–275, 1998.

A. Rényi. On random generating elements of a finite Boolean algebra. *Acta Sci. Math. (Szeged)*, 22:75–81, 1961.

I. T. Roberts. The flat antichain conjecture for small average set size. Technical report, Northern Territory University, 1999.

I. T. Roberts and Leanne J. Rylands. Minimal (n) and (n,h,k) completely separating systems. *Australasian Journal of Combinatorics*, 33:57 – 66, 2005.

I. Shpitser and J. Pearl. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, 2006.

J. Spencer. Minimal completely separating systems. *Journal of Combinatorial Theory*, 8(4):446 – 447, 1970.

E. Sperner. Ein Satz über Untermengen einer endlichen Menge. *Mathematische Zeitschrift*, 27: 544–548, 1928.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search*. MIT Press, 1993.

S. Tong and D. Koller. Active learning for structure in Bayesian networks. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, 2001.

K. Watanabe, M. Sengoku, H. Tamura, K. Nakano, and S. Shinoda. A scheduling problem in multihop networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E83-A(6):1222 – 1227, 2000.

I. Wegener. On separating systems whose elements are sets of at most k elements. *Discrete Mathematics*, 28(2):219 – 222, 1979.

D. J. A. Welsh and M. B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86, 1967.

Journal of Machine Learning Research 14 (2013) 3073-3104 Submitted 5/12; Revised 5/13; Published 10/13

Stationary-Sparse Causality Network Learning

Yuejia He YUEJIAHE@UFL.EDU
Department of Electrical and Computer Engineering
University of Florida
Gainesville, FL 32611-6130

Yiyuan She YSHE@STAT.FSU.EDU
Department of Statistics
Florida State University
Tallahassee, FL 32306-4330

Dapeng Wu WU@ECE.UFL.EDU
Department of Electrical and Computer Engineering
University of Florida
Gainesville, FL 32611-6130

Editor: Hui Zou

Abstract

Recently, researchers have proposed penalized maximum likelihood to identify network topology underlying a dynamical system modeled by multivariate time series. The time series of interest are assumed to be stationary, but this restriction is never taken into consideration by existing estimation methods. Moreover, practical problems of interest may have ultra-high dimensionality and obvious node collinearity. In addition, none of the available algorithms provides a probabilistic measure of the uncertainty for the obtained network topology which is informative in reliable network identification. The main purpose of this paper is to tackle these challenging issues. We propose the $\mathbf{S}^2$ learning framework, which stands for *stationary-sparse* network learning. We propose a novel algorithm referred to as the Berhu iterative sparsity pursuit with stationarity (BISPS), where the Berhu regularization can improve the Lasso in detection and estimation. The algorithm is extremely easy to implement, efficient in computation and has a theoretical guarantee to converge to a global optimum. We also incorporate a screening technique into BISPS to tackle ultra-high dimensional problems and enhance computational efficiency. Furthermore, a stationary bootstrap technique is applied to provide connection occurring frequency for reliable topology learning. Experiments show that our method can achieve stationary and sparse causality network learning and is scalable for high-dimensional problems.

Keywords: stationarity, sparsity, Berhu, screening, bootstrap

1. Introduction

There has been an increasing interest in identifying network dynamics and topologies in the emerging scientific discipline of network science (e.g., Newman and Watts, 2006; Lewis, 2009). In a dynamical network, the evolution of a node is controlled not only by itself, but also by other nodes. For example, in the gene regulatory network (Faith et al., 2007), the expression levels of genes influence each other, following some dynamic rules, which connect the genes together and form a dynamical system. If the topology and evolution rules of the network are known, we can analyze the

regulation between genes or detect unusual behaviors to help diagnose and cure genetic diseases. Similarly, the modeling and estimation of dynamical networks are of great importance for various domains including stock market (Mills and Markellos, 2008), brain network (Bullmore and Sporns, 2009) and social network (Hanneke et al., 2010). To accurately identify the topology and dynamics underlying those networks, scientists are devoted to developing appropriate mathematical models and corresponding estimation methods.

In practice, we can obtain discrete observations of the network over a period of time, which can usually be modeled by multivariate time series from a statistical perspective. For example, the fMRI data of the human brain is taken every one minute during a two-hour experiment; stock prices are often recorded daily or weekly. These multivariate time series contain important information of the network topology and dynamics. Vector autoregressive (VAR) process (Sims, 1980) is one of the most commonly used models for characterizing the relations between the time series. In this model, the state of each node is characterized by a time series. The value of a node at a time point is a linear combination of the past values of itself and the nodes regulating it. This kind of regulation relationship is regarded as the *Granger causal connection* (Granger, 1969). By estimating the transition matrix of the model, we can understand the Granger causal relations between nodes.

In estimating the transition matrix, one must bare in mind two most important objectives: first, the estimate fitted on the training data should provide accurate prediction; second, a sparse topology that illustrates the most prominent network connections is desired. Recently, compressive sensing approaches based on penalized maximum likelihood (PML) are applied to achieve accurate prediction and sparse representation simultaneously (Donoho, 2006; Tsaig and Donoho, 2006; Songsiri and Vandenberghe, 2010). Different penalties and algorithms are proposed (Fan and Li, 2006, 2001; Zou, 2006; Zou and Hastie, 2005; Blumensath and Davies, 2010). The ℓ_1 penalty (Tibshirani, 1996) is popular for its computational efficiency and theoretical elegance. Nevertheless, the major problem of the ℓ_1 penalty for dynamical network learning is its incapability of handling collinearity, which typically exists in network data as a result of the interaction between nodes. The elastic net (Zou and Hastie, 2005) uses a linear combination of the ℓ_1 and ℓ_2 penalties to deal with collinearity and large noise. However, its ℓ_2 component may counteract sparsity and bring the so-called "double shrinkage" issue. To improve these drawbacks, we study a new '$\ell_1 + \ell_2$' variant—Berhu (Owen, 2007), which fuses the ℓ_1 and ℓ_2 penalties in a nonlinear fashion and thus can deal with collinearity as well as achieve sufficient sparsity. We propose a Berhu thresholding operator to efficiently solve the Berhu penalized problem.

In real-world problems, raw observations from a dynamical network are usually preprocessed and stationarized before the application of PML (Stock and Watson, 2012; Hsu et al., 2008). Nevertheless, as will be demonstrated in the experiment, it is possible for PML to end up with a nonstationary estimate, due to noise contamination and limited number of observations. Such nonstationary estimates may give unmeaningful prediction results, especially for **long-term** forecasting. Hence, our work, distinguished from the existing ones, focuses on enforcing the stationarity guarantee in network estimation and topology identification, which is of great importance but has never been properly addressed. The stationarity condition of the VAR model is its spectral radius being smaller than one (Reinsel, 1997). This constraint is nonconvex and extremely difficult to tackle directly (Burke et al., 2005; Curtis and Overton, 2012; Overton and Womersley, 1988). Hence, we use a convex relaxation and come up with a stationarity constrained PML problem. We propose an efficient algorithm, the *Berhu iterative sparsity pursuit with stationarity* (BISPS), to achieve *stationary-sparse* ($\mathbf{S}^2$) network learning. This algorithm is very easy to implement and theoretically

guaranteed to converge to a global optimum. Experimentation demonstrates that our method can guarantee a stationary and sparse estimate. It not only gives satisfactory identification accuracy, but also outperforms the plain PML method significantly in prediction.

Another challenge in network identification lies in the high dimensionality of the data (Fan and Lv, 2010). For a network with p nodes and n observations, the number of unknown variables in the transition matrix is p^2, and we frequently face practical data sets with $p^2 \gg n$, for example, microarray data sets consisting of thousands of genes but fewer than a hundred observations. This so-called "ultra-high dimensional" problem (Fan and Lv, 2008) adds tremendous difficulties to the inference methods in terms of statistical estimation accuracy as well as computational complexity. To address this challenging issue, we propose two efficient techniques. First, the *quantile thresholding iterative screening* (QTIS) is designed to "preselect" connections for the BISPS algorithm in a supervised manner. QTIS differs from existing screening techniques such as the sure independence screening (Fan and Lv, 2008) in that it takes into account of collinearity in the data. Secondly, we propose the *stationary bootstrap* enhanced BISPS (SB-BISPS). Bootstrap is a nonparametric technique for approximating the distributions of statistics or constructing confidence intervals. Our work applies this powerful tool with stationarity guarantee to network identification and provides a confidence level for the occurrence of each possible connection in the network.

The remainder of the paper is organized as follows: Section 2 introduces the stationary and sparse network model and formulates the $\mathbf{S}^2$ learning framework. Section 3 proposes our algorithms, mainly the Berhu iterative sparsity pursuit with stationarity (BISPS) and the quantile thresholding iterative screening (QTIS), and provides theoretical proofs for their convergence. Section 4 describes the stationary bootstrap enhanced BISPS (SB-BISPS). In Section 5, we show experimental results on synthetic data. In Section 6, we apply the proposed method to the U.S. macroeconomic data. Section 7 concludes our work.

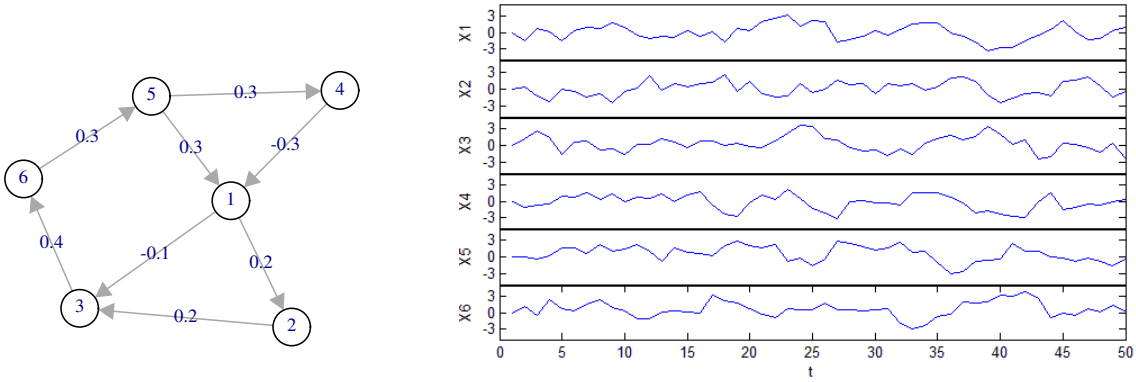
2. The Stationary-Sparse (S^2) Network Learning Framework

Let x be a p-dimensional random vector with each component being a time series associated with one node in a dynamical network, where p is the number of nodes. We are interested in characterizing the observations of x at different time points using mathematical models, based on which we can conduct useful network analysis. In particular, we are interested in understanding the causal relations between nodes and making predictions for future. A commonly used model describes the current state x_t of the system as a linear transformation of its previous state x_{t-1}:

$$x_t = Bx_{t-1} + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \Sigma_\varepsilon), \tag{1}$$

where B is the *transition matrix* and ε_t is random noise. This corresponds to the first-order vector autoregressive (VAR) model (Sims, 1980). It can be generalized to a VAR model with order m, where the current state is a linear combination of the most recent m states. On the other hand, any mth-order VAR model can be converted to a first-order VAR model by appropriately redefining the node variables (Lütkepohl, 2007), and thus we focus on the former one with $m = 1$ in this paper.

In (1), the transition matrix $B = [b_{ij}]_{1 \le i,j \le p}$ describes a network that represents the dynamical system: if $b_{ij} \neq 0$, there is a *Granger causal connection* (Granger, 1969) from node j to node i with weight b_{ij}. In other words, node j Granger-causes node i. For example, for a network with 6 nodes



(a) causality network

(b) multivariate time series

Figure 1: Example of a network (1) with transition matrix (2).

and a transition matrix as

$$B = \begin{pmatrix} 0.6 & 0.2 & -0.1 & 0 & 0 & 0 \\ 0 & 0.6 & 0.2 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 & 0.4 \\ -0.3 & 0 & 0 & 0.7 & 0 & 0 \\ 0.3 & 0 & 0 & 0.3 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0.3 & 0.6 \end{pmatrix}, \tag{2}$$

Figure 1a shows its topology (self-connections are removed). The nodes evolve and interact with each other through the Granger causal connections, resulting in the random processes plotted in Figure 1b. Therefore, matrix B not only illustrates the dynamic rules that govern the evolution of the system, but also captures a linear *causality network* that describes the (Granger) casual relations between nodes.

2.1 Sparse Network Learning by Penalized Maximum Likelihood Estimation

Given n observations of the dynamical network $x_1, \cdots, x_n$, we wish to estimate B. Due to Markov-chain property, we can write the likelihood of B as

$$L(B|x_1, \cdots, x_n) = \prod_{t=2}^{n} f(x_t|x_{t-1}, \cdots, x_1, B) f(x_1|B) = \prod_{t=2}^{n} f(x_t|x_{t-1}, B) f(x_1|B).$$

The exact maximum likelihood (ML) estimate requires solving a nonlinear optimization problem. For simplicity, researchers often use the *conditional* likelihood where the initial state x_1 is assumed to be fixed. Due to normality, we have the conditional likelihood

$$L_c(B) = \prod_{t=2}^{n} f(x_t|x_{t-1}, B) = \prod_{t=2}^{n} (2\pi)^{-p/2} |\Sigma_\varepsilon|^{-1/2} \exp\{-\frac{1}{2}(x_t - Bx_{t-1})^\top \Sigma_\varepsilon^{-1} (x_t - Bx_{t-1})\}.$$

So the (conditional) ML estimate of B can be obtained by solving

$$\min_B \frac{1}{2} \sum_{t=2}^{n} \|x_t - Bx_{t-1}\|_2^2.$$

Letting $Y = [x_2^\mathsf{T}, x_3^\mathsf{T}, \cdots, x_n^\mathsf{T}]^\mathsf{T}$, $X = [x_1^\mathsf{T}, x_2^\mathsf{T}, \cdots, x_{n-1}^\mathsf{T}]^\mathsf{T}$ and $A = B^\mathsf{T}$, we can formulate the problem in matrix form:

$$A_{ML} = \arg\min_A l(A) = \frac{1}{2}\|Y - XA\|_F^2.$$

For convenience, we use A instead of B to represent the network in the remainder of the paper. Note that a_{ij} describes the directed connection strength from node i to node j. The estimate $\hat{A}_{ML}$ has been investigated and applied to many real-world data. For stationary process, the consistency and asymptotic efficiency of $\hat{A}_{ML}$ are analyzed in Reinsel (1997). The small-sample properties are discussed in Lütkepohl (2007).

Nevertheless, the plain ML estimation is not ideal for network learning. In practice, X usually demonstrates high collinearity, especially when some nodes have similar dynamical behaviors and when the number of observations is limited. Moreover, the ML estimation does not promote sparsity and consequently the resulting model is difficult to interpret. To improve prediction accuracy and obtain interpretable model, *shrinkage estimation* is necessary. It can be done by adding a penalty and/or constraint. For example, we can estimate A via penalized maximum likelihood (PML):

$$\hat{A}_{PML} = \arg\min_A l(A) + P(A;\lambda). \tag{3}$$

We consider only the additive penalties and denote $P(A;\lambda) = \sum_{i,j} P(a_{ij};\lambda_{ij})$, where $P(\cdot)$ is a penalty function applied to each component of A, and λ_{ij} is the corresponding regularization parameter(s). Alternatively, constraints can also be used. See Section 2.3 and Section 3.4.

Different penalties have been proposed. The famous Lasso (Tibshirani, 1996) solves the ℓ_1 penalized problem. It is fast in computation. Nevertheless, Lasso suffers from some drawbacks such as selection inconsistency, estimation bias and incapability of dealing with collinearity, in particular. Zou and Hastie (2005) propose the elastic net (eNet for short in this paper) which adds an additional ridge regularization (Hoerl and Kennard, 1970) to deal with collinearity and large noise. However, the design counteracts sparsity to some extend and may bring the double shrinkage issue. Some nonconvex alternatives, including the '$\ell_0 + \ell_1$' SCAD (Fan and Li, 2001) and the '$\ell_0 + \ell_2$' hard-ridge (She, 2009, 2012), are advocated to promote more sparsity. However, due to nonconvexity, the convergent solution may be only locally optimal and depend on the choices of the initial point. They are also more computationally expensive than convex approaches. Therefore, we do not consider nonconvex regularizations hereinafter.

2.2 The S^2 Network Learning

Many real-world time series (possibly after proper transformations such as taking logarithm and/or differencing) are stationary. Stationary and nonstationary processes behave in fundamentally different manners. See Figure 2. For a stationary process, its probability distribution is invariant with respect to the shift in time. In the nonstationary process, however, we can clearly see drifting and trending behaviors. In practice, given raw observations sampled from a dynamical system, researchers first stationarize the time series and then input them to the ML/PML estimator. The resulting estimate $\hat{A}_{ML}/\hat{A}_{PML}$ is used for analysis and forecast. Unfortunately, however, the stationarity requirement may be violated by $\hat{A}_{ML}/\hat{A}_{PML}$ in practice. Figure 3 shows a real-data example. We apply ML and PML (adopting the ℓ_1 penalty) respectively to the U.S. macroeconomic data (Stock and Watson, 2012), which are stationary after proper transformations. The estimates are then used to forecast an index "GDP263". As shown in Figure 3, though the time series of GDP263

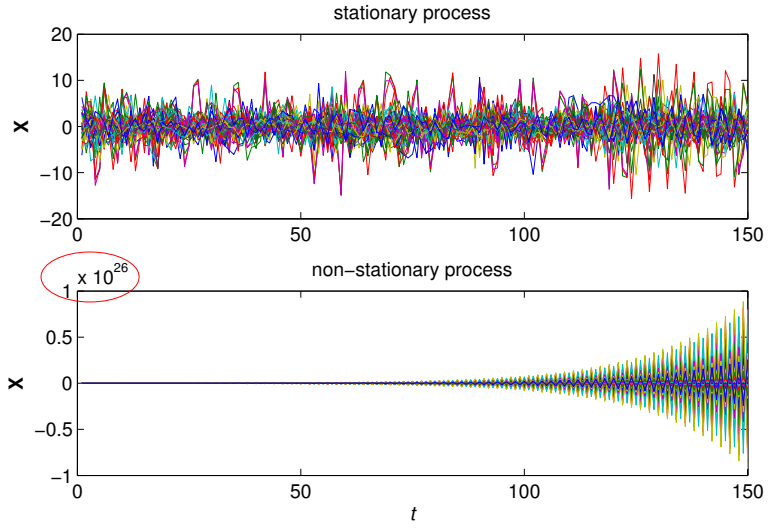


Figure 2: Example of stationary and nonstationary processes. The number of nodes is $p = 50$. The stationary process has $\rho(A) = 0.95$ and the nonstationary process has $\rho(A) = 1.05$.

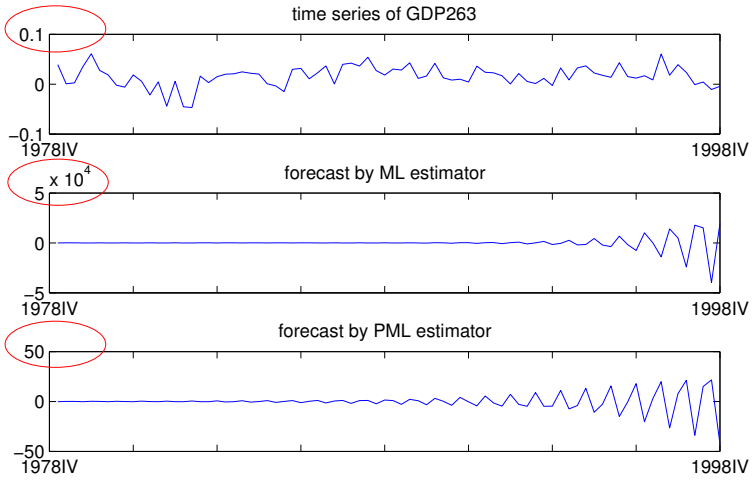


Figure 3: Forecasts of GDP263 given by ML and PML estimation. The time series of GDP263 is obtained from seasonal observations between 1978:IV and 1998:IV. It is a stationary process. However, the forecasts given by ML and PML exhibit nonstationary behaviors. $\rho(\hat{A}_{ML}) = 1.171$, $\rho(\hat{A}_{PML}) = 1.073$.

is stationary, the ML and PML forecasts clearly exhibit nonstationary behaviors and they fail to capture all characteristics of the original time series.

In this paper, we propose the framework of *stationary-sparse* ($\mathbf{S}^2$) network learning to address the limitation of PML to guarantee the stationarity property of the network. We invoke the stationarity condition and design an efficient algorithm to solve the optimization problem.

A random process as in (1) is stationary if and only if its spectral radius $\rho(A)$ satisfies the *stationarity condition*:

$$\rho(A) \stackrel{\Delta}{=} \max_i |\lambda_i| < 1, \tag{4}$$

where λ_i is the ith eigenvalue of A, possibly complex, and $|\cdot|$ is the complex norm (Reinsel, 1997). This leads to the following optimization problem:

$$\begin{aligned} \min_A \ & f(A) = \frac{1}{2}\|Y - XA\|_F^2 + P(A;\lambda) \\ & \text{s.t. } \rho(A) < 1. \end{aligned} \tag{5}$$

Nevertheless, problem (5) is extremely challenging due to the fact that $\rho(A)$ is a nonconvex and non-Lipschitz-continuous function of A. An optimization method proposed by Curtis and Overton (2012), which combines sequential quadratic programming and gradient sampling, sheds some light on solving (5). However, at each iteration, the gradient sampling needs to sample p^2 points and calculate the gradient of the spectral radius at each point. As discussed in Overton and Womersley (1988), calculating the gradient of spectral radius for a single point is already a challenging and computationally demanding problem. It is prohibitive to do so for p^2 points at each iteration in our problem. Moreover, this method only guarantees $\rho(A) \leq 1$, not $\rho(A) < 1$.

We consider a reasonable convex relaxation of (4) as the stationarity constraint:

$$\|A\|_2 \stackrel{\Delta}{=} \max_i |\nu_i| \leq 1,$$

where ν_i is the ith singular value of A and thus $\|A\|_2$ is the spectral norm. For an arbitrary square matrix, we have $\rho(A) \leq \|A\|_2$, where the equality holds when A is a symmetric matrix. In all our applications, we have $\rho(A) < 1$.

The $\mathbf{S}^2$ learning problem is given by

$$\begin{aligned} \hat{A}_{S^2} = \arg\min_A \ & f(A) \\ & \text{s.t. } \|A\|_2 \leq 1. \end{aligned} \tag{6}$$

Note that the stationarity constraint also has a "shrinking" effect on the estimate, which contributes to the shrinkage estimation we are seeking, as discussed in Section 2.1.

2.3 The "Berhu" Penalty for Sparsity Pursuit and Model Decorrelation

As discussed in Section 2.1, coherence is often observed in real-world network data, especially when some nodes have similar dynamical behaviors or strong influence between each other. In such cases, the conventional Lasso fails to handle collinearity and consequently gives unsatisfactory identification and forecasting performance. Hence, a more proper penalty is in need for $\mathbf{S}^2$ network learning. We adopt a new hybrid penalty "Berhu", which has a close relation with Huber's loss function for robust regression (Huber, 1981). The Huber function is quadratic at small values and

linear at large ones, which makes it more robust to outliers than the squared-error criterion. Inspired by the Huber function, Owen (2007) designed a convex penalty function Berhu

$$P_{\mathcal{B}}(t;\lambda,M)=\begin{cases}\lambda|t| & \text{if } |t|\le M\\ \lambda\frac{t^2+M^2}{2M} & \text{if } |t|>M.\end{cases} \tag{7}$$

As implied by its name, Berhu reverses the composition of Huber: it is linear at small values and quadratic at large ones.

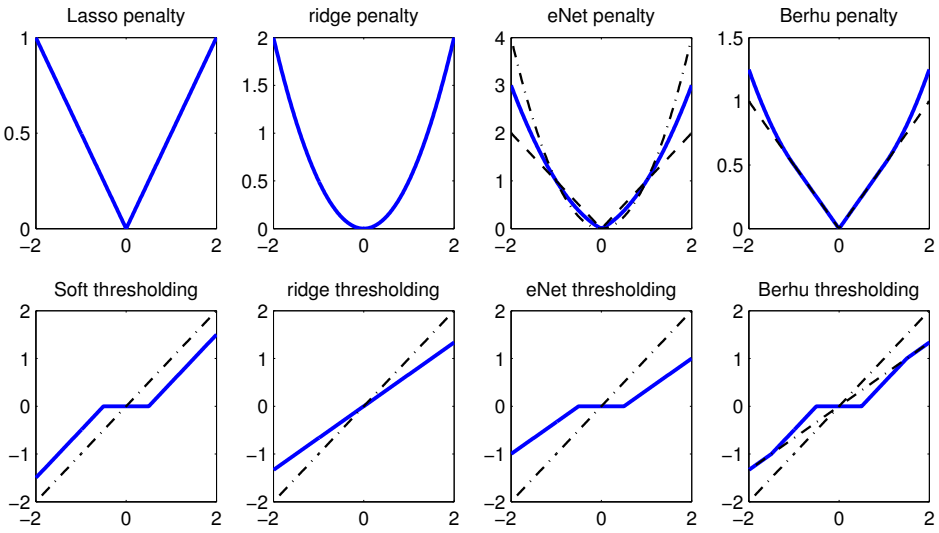


Figure 4: Penalty functions and corresponding solutions.

Figure 4 compares Berhu with the ridge penalty $P_R(t;\eta)=\frac{1}{2}\eta t^2$, Lasso $P_L(t;\lambda)=\lambda|t|$ and eNet $P_E(t;\lambda,\eta)=\lambda|t|+\frac{1}{2}\eta t^2$. The upper panel plots the functions, while the lower panel shows their corresponding solutions in the univariate and orthogonal case (see Section 3.2 for details). The ridge penalty shrinks the coefficients to compensate for collinearity. But it can not produce exact zero coefficients in the estimate. The Lasso soft-thresholds the coefficients to encourage sparsity. However, it does not shrink the large coefficients effectively and does not work well for correlated data. The eNet incorporates the ridge component into the ℓ_1. However, the singularity of the penalty function at zero is smoothed out to some extent by the ℓ_2 part, which may lead to an estimate not parsimonious enough. Also, it tends to over-shrink medium and large coefficients (Zou and Hastie, 2005). Berhu overcomes these drawbacks by using a nonlinear fusion of the ℓ_1 and ℓ_2 penalties: for small coefficients, the ℓ_1 regularization is enforced to achieve sparsity; for large coefficients, the ℓ_2 regularization is enforced to compensate collinearity (Hoerl and Kennard, 1970) and multi-dimensionality (James and Stein, 1961). As a result, Berhu not only preserves the singularity property of Lasso at zero but also inherits the advantage of ridge regression in model decorrelation. It is convex as well. The difference between Berhu and eNet is significant. For medium and large coefficients, eNet not only shifts but also shrinks, which results in the double shrinkage effect (Zou and Hastie, 2005). On the other hand, Berhu shifts only medium coefficients

and shrinks large ones. It does selection and decorrelation separately, which better serves two important objectives of network learning: accurate prediction and parsimonious representation.

Substituting $P_{\mathcal{B}}(A;\lambda,M)$ into (6), we focus on solving

$$\begin{aligned} \arg\min_{A} \ f_{\mathcal{B}}(A) = \frac{1}{2}\|Y - XA\|_F^2 + P_{\mathcal{B}}(A;\lambda,M) \\ \text{s.t. } \|A\|_2 \leq 1. \end{aligned} \tag{8}$$

In this problem, $P_{\mathcal{B}}(A;\lambda,M)$ is typically nondifferentiable at zero and piecewise. The stationarity constraint adds more difficulties to the problem. Moreover, in practice we are frequently confronted with large-scale networks. Hence, an efficient and scalable algorithm is desired for $\mathbf{S}^2$ network learning.

3. Computation of BISPS

In this section, we propose an algorithm named *Berhu iterative sparsity pursuit with stationarity* (BISPS) to effectively solve the $\mathbf{S}^2$ learning problem (8). Some algorithms based on conventional techniques will be developed first. They suffer from high computational complexity, poor numerical accuracy, and/or insufficient sparsity. We then propose the novel BISPS which is easy to implement and computationally efficient. Finally, to facilitate BISPS for ultra-high dimensional problems, we propose the *quantile thresholding iterative screening* (QTIS). Convergence proofs are provided. We assume the data matrices X, Y has been centered before all the computation. Precalculations $\Sigma_{XX} \triangleq X^\top X$ and $\Sigma_{XY} \triangleq X^\top Y$ help avoid repeated computation.

3.1 Algorithms Based on Conventional Techniques

Problem (8) can be reformulated and then solved by well-known optimization techniques, such as semidefinite programming, projected subgradient method, and alternating direction method of multipliers. We briefly discuss these algorithms before introducing BISPS.

3.1.1 SEMIDEFINITE PROGRAMMING

Problem (8) can be reformulated as a semidefinite programming (SDP) problem

$$\begin{aligned} &\min_{A} f_{\mathcal{B}}(A) \\ \text{s.t. } &\begin{bmatrix} I & A \\ A^\top & I \end{bmatrix} \succeq 0 \end{aligned}$$

and can be solved by general SDP solvers. However, since most of the SDP solvers use interior point methods, they suffer from extremely high space complexity. For example, we tried the popular SDP solvers SeDuMi (Sturm, 1998) and SDPT3 (Tütüncü et al., 2003) using MATLAB7.11.0 on a PC with 4GB memory; when the number of network nodes is larger than 100, both solvers ran out of memory.

3.1.2 PROJECTED SUBGRADIENT METHOD

Define the subgradient of $f_{\mathcal{B}}(A)$ at A as

$$\partial f_{\mathcal{B}}(A) = \nabla l(A) + \partial P_{\mathcal{B}}(A;\lambda,M),$$

where $\partial P_{\mathcal{B}}(A;\lambda,M)$ is the subdifferential (Alber et al., 1998) of $P_{\mathcal{B}}(\cdot)$, and $\nabla l(A)$ is the gradient of $l(A)$: $\nabla l(A) = \Sigma_{XX}A - \Sigma_{XY}$. The projected subgradient method (PSGM) for problem (8) computes a sequence of feasible points $\{A^k\}$ with the update rule

$$A^{k+1} = \Pi(A^k - \alpha_k U^k; 1), \text{ where } U^k \in \partial f_{\mathcal{B}}(A^k),$$

until A^k satisfies $0 \in \partial f_{\mathcal{B}}(A^k)$. The operator Π stands for spectral norm projection defined in Lemma 5.

PSGM is simple to implement. At each step, one performs subgradient evaluation and spectral norm projection. Nevertheless, due to the uncertainty of the subgradient at the non-differentiable point of the penalty function, it suffers from slow convergence and insufficiency of sparsity. For example, in an experiment where the number of nodes $p = 100$ and number of observations $n = 80$, it does not converge yet after 10^4 iterations.

3.1.3 ALTERNATING DIRECTION METHOD OF MULTIPLIERS

The basic idea of alternating direction method of multipliers (ADMM) is to split the objective function and variables and update them in an alternating fashion (Boyd et al., 2010). To apply ADMM for solving problem (8), we reformulate it as

$$\begin{aligned} \min_{A,B,C} \frac{1}{2}\|Y - XA\|_F^2 + P_{\mathcal{B}}(B;\lambda,M) \\ \text{s.t. } \begin{bmatrix} A \\ A \end{bmatrix} = \begin{bmatrix} B \\ C \end{bmatrix}, \\ \text{and } \|C\|_2 \le 1. \end{aligned}$$

The augmented Lagrangian can then be written as $L_{\rho_1,\rho_2}(A,B,C,\Gamma_1,\Gamma_2) = \frac{1}{2}\|Y - XA\|_F^2 + P_{\mathcal{B}}(B;\lambda,M) + \mathrm{tr}\{\Gamma_1^{\mathsf{T}}(A-B)\} + \mathrm{tr}\{\Gamma_2^{\mathsf{T}}(A-C)\} + \frac{\rho_1}{2}\|A-B\|_F^2 + \frac{\rho_2}{2}\|A-C\|_F^2$, where Γ_1 and Γ_2 are Lagrangian multipliers and ρ_1 and ρ_2 are the augmented Lagrangian parameters. The iteration of ADMM consists of the following steps

$$\begin{aligned} A^{k+1} &= (\Sigma_{XX} + \rho_1 I + \rho_2 I)^{-1}(\Sigma_{XY} - \Gamma_1^k - \Gamma_2^k + \rho_1 B^k + \rho_2 C^k), \\ B^{k+1} &= \Theta_{\mathcal{B}}(A^{k+1} + \Gamma_1^k/\rho_1; \lambda/\rho_1, M), \\ C^{k+1} &= \Pi(A^{k+1} + \Gamma_2^k/\rho_2; 1), \\ \Gamma_1^{k+1} &= \Gamma_1^k + \rho_1(A^{k+1} - B^{k+1}), \\ \Gamma_2^{k+1} &= \Gamma_2^k + \rho_2(A^{k+1} - C^{k+1}), \end{aligned}$$

where $\Theta_{\mathcal{B}}$ is the thresholding operator of Berhu—see Appendix A for detail. Note that matrix inversion is involved in updating A, which increases computational difficulty. The penalty parameters ρ_1 and ρ_2 have to be large enough; the choices of them have been shown to influence the number of iterations significantly. To speed up convergence in practice, one can replace the constants ρ_1 and ρ_2 with two sequences $\{\rho_1^k\}$ and $\{\rho_2^k\}$ respectively, where ρ_1^k and ρ_2^k vary along the iterations following some *ad hoc* adaptive rule (He et al., 2000). However, the algorithm is still slow when the problem is high dimensional. For example, when $p = 300, n = 100$, ADMM costs up to 10 times more computation time than BISPS (to be described in Section 3.2) to reach comparable accuracy. Also, the convergence property of ADMM with varying ρ is not clear.

In summary, although we have implemented some algorithms based on conventional techniques for the $\mathbf{S}^2$ network learning problem, they are unable to cope with large-scale networks. A more efficient and scalable algorithm is in great need.

3.2 The Berhu Thresholding Operator

Section 2.3 advocates the Berhu penalty for $\mathbf{S}^2$ network learning. However, in the original paper (Owen, 2007), Berhu was solved through cvx (Grant and Boyd, 2008). Practical applications call for the development of much faster algorithms. In this paper, we reparameterize Berhu and develop its coupled *thresholding rule*, which allows us to solve the Berhu sparsity pursuit—problem (3) with Berhu penalty—in a simple and efficient way. This formulation facilitates easy parameter tuning. Also, it helps us understand the essence of Berhu.

Let $\eta = \lambda/M$. Reformulate the Berhu penalty (7) as

$$P_{\mathcal{B}}(t;\lambda,\eta) = \begin{cases} \lambda|t| & \text{if } |t| \le \lambda/\eta \\ \frac{\eta^2 t^2+\lambda^2}{2\eta} & \text{if } |t| > \lambda/\eta. \end{cases} \tag{9}$$

Define a thresholding rule

$$\Theta_{\mathcal{B}}(t;\lambda,\eta) = \begin{cases} 0 & \text{if } |t| < \lambda \\ t-\lambda\text{sgn}(t) & \text{if } \lambda \le |t| \le \lambda+\lambda/\eta \\ \frac{t}{1+\eta} & \text{if } |t| > \lambda+\lambda/\eta. \end{cases} \tag{10}$$

It can be verified that, as shown in Lemma 3, $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is the coupled thresholding rule for the Berhu penalty $P_{\mathcal{B}}(\cdot;\lambda,\eta)$:

$$P_{\mathcal{B}}(t;\lambda,\eta) = \int_0^{|t|} (\sup\{s : \Theta_{\mathcal{B}}(s;\lambda,\eta) \le u\} - u)du.$$

For the multivariate case, the Berhu thresholding operator is applied elementwise.

With $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$, we can solve the Berhu sparsity pursuit (without the stationarity constraint) using a simple iterative procedure:

$$A^{k+1} = \Theta_{\mathcal{B}}(A^k + \alpha_k(\Sigma_{XY} - \Sigma_{XX}A^k);\lambda,\eta). \tag{11}$$

Since $P_{\mathcal{B}}(\cdot;\lambda,\eta)$ is convex, algorithm convergence is guaranteed given $\alpha_k \le \sqrt{2}/\|X\|_2$ (She, 2012).

It is worth pointing out that, based on the construction rule (13), we can also define the thresholding operators for other penalties including Lasso, eNet and the ridge penalty, as shown in Figure 4. The Berhu thresholding operator $\Theta_{\mathcal{B}}(t;\lambda,\eta)$ offers a nonlinear fusion of the soft thresholding operator (coupled with Lasso) $\Theta_S(t;\lambda) = \text{sgn}(t)(|t|-\lambda)1_{|t|\ge\lambda}$ and the ridge thresholding operator $\Theta_R(t;\eta) = \frac{t}{1+\eta}$. For the difference between the Berhu thresholding and the eNet thresholding $\Theta_E(t;\lambda,\eta) = \frac{1}{1+\eta}\text{sgn}(t)(|t|-\lambda)1_{|t|\ge\lambda}$, see the discussion in Section 2.3.

3.3 The BISPS Algorithm

Based on the Berhu thresholding operator (10), we now propose BISPS as given in Algorithm 1. This algorithm contains only simple matrix operations in addition to the *spectral norm projection*

Algorithm 1 The Berhu iterative sparsity pursuit with stationarity (BISPS)

Input: data matrix Σ_{XX}, Σ_{XY}; regularization parameters λ, η; stopping criteria δ_1,δ_2, M_1, M_2; initial estimate A^0.

{Let $\|\cdot\|_2$ denotes the spectral norm and $\|\cdot\|_{\max}$ denote the elementwise max-norm.}

$k_0 \leftarrow$ any constant satisfying $k_0 > \|X\|_2$;

$k \leftarrow 0$;

repeat

1) $B^0 \leftarrow A^k + \frac{1}{k_0^2}(\Sigma_{XY} - \Sigma_{XX}A^k)$;

2) $j \leftarrow 0; P^0 \leftarrow 0; Q^0 \leftarrow 0$;

repeat

2.1) $C^j = \Theta_{\mathcal{B}}(B^j + P^j; \lambda/k_0^2, \eta/k_0^2)$;

2.2) $P^{j+1} = B^j + P^j - C^j$;

2.3) $B^{j+1} = \Pi(C^j + Q^j; 1)$;

2.4) $Q^{j+1} = C^j + Q^j - B^{j+1}$;

$j \leftarrow j+1$;

until $\|B^j - B^{j-1}\|_{\max} \leq \delta_2$ or $j \geq M_2$

3) $A^{k+1} \leftarrow B^j$;

$k \leftarrow k+1$;

until $\|A^k - A^{k-1}\|_{\max} \leq \delta_1$ or $k \geq M_1$

$\hat{A} \leftarrow A^k$;

Output: $\hat{A}$.

Π. Parameter k_0 can be set to any constant that is larger than the spectral norm of X. No *ad hoc* algorithmic parameters, such as ρ_1, ρ_2 in ADMM and α_k in PSGM, are involved. The inner iteration of Step 2 often converges within 10 steps in practice, where matrices C, P, Q are auxiliary variables that contribute to fast convergence of the procedure. Step 2.1 is to enforce sparsity by Berhu thresholding and Step 2.3 is to project the estimate to the convex set $\{B : \|B\|_2 \leq 1\}$. The outer iteration has only a simple update step and converges fast. As a result, the algorithm is computationally efficient as well as easy to implement.

The convergence of BISPS is theoretically guaranteed. For simplicity, we assume the inner iteration is run till convergence. Theorem 1 states that Algorithm 1 solves the $\mathbf{S}^2$ network learning problem.

Theorem 1 *Suppose* $\lambda \geq 0, \eta \geq 0$ *and* $\lambda\eta \neq 0$. *Given* $k_0 > \|X\|_2$, *for any initial value* A^0, *the sequence of iterates* $\{A^k\}$ *produced by Algorithm 1 converges to a* ***globally*** *optimal solution to problem* (8).

See Appendix A for the detailed proof.

BISPS has more flexibility and generality. Though it is designed with the Berhu penalty, by replacing $\Theta_{\mathcal{B}}$ with an appropriate thresholding operator in Step 2.1, the algorithm allows for any convex penalty for $\mathbf{S}^2$ learning. Moreover, if Step 2.2 to Step 2.4 are removed, Algorithm 1 reduces to the Berhu sparsity pursuit (11).

The most expensive computation of Algorithm 1 lies in the spectral norm projection (Step 2.3). In practice, we can apply some techniques to further improve computational efficiency. 1) We can first run Algorithm 1 without Step 2.2 to Step 2.4 and obtain an estimate. If it satisfies the

stationarity condition (4), we accept and output this solution. Otherwise, we rerun Algorithm 1 with Step 2.2 to Step 2.4 included. 2) Moreover, we can take advantage of the fact that A^{k+1} is sparse and the number of singular values of A^{k+1} that are larger than 1 is much smaller than p. The singular value thresholding algorithm (Cai et al., 2010), among some other fast algorithms, calculates only the singular values that are above a threshold and their corresponding singular vectors, which is computationally efficient for large sparse matrix and thus fits our problem well. We use the package MODIFIED-PROPACK provided by Lin (2011) to calculate the partial SVD with threshold being 1. For $p = 500, n = 100$, the partial SVD, compared with the original SVD, can accelerate the calculation by up to 30 times.

3.4 Quantile Thresholding Iterative Screening

Nowadays, a great challenge for network identification and statistical learning comes from the large scale of the system. For example, for a network with $p = 1000$ nodes, the number of variables in the transition matrix is as large as $p^2 = 10^6$, which poses a great challenge for any estimation algorithm in scalability and stability. As a result, ultra-high dimensional learning has become a hot topic (Fan et al., 2009; Fan and Lv, 2010) . For regression problems, under the assumption that the number of nonzero coefficients is far smaller than n, *screening* techniques can be used to coarsely select the variables before finer estimation. This idea can be adopted in network identification: if one is sure that the average number of connections for each node is much less than $\lceil \mu n \rceil$ (say $\mu = 0.8$) or the total number of connections in the network is much less than $\lceil \mu pn \rceil$, one can first use fast screening techniques to select $m = \lceil \mu pn \rceil$ candidate connections, and then apply BISPS restricted on the candidate connections for further selection and estimation. If the screening technique can include all the true connections with high probability, dramatic computational gain can be attained with mild performance sacrifice.

Independence screening methods, such as the sure independence screening (SIS) (Fan and Lv, 2008) can be applied to preselect variables in a supervised manner. Applied to network learning, SIS sorts the elements of $W = X^\top Y$ by magnitude in a decreasing order and defines a reduced model

$$\mathcal{M}_\mu = \{(i, j) : |w_{ij}| \text{ is among the } m \text{ largest of all}, 1 \le i, j \le p\}.$$

This method is simple and fast, but it relies on the assumption that the predictors are *independent*, since it only studies the marginal correlation between Y and X and selects the variables accordingly. In network settings, the nodes are interacting dynamically with each other, so there is usually high collinearity in the data. In such cases, SIS is too greedy and misses many true connections.

To derive a new screening technique that can handle network data, we first observe that SIS corresponds to the first step of the iterative procedure in (11) with $A^0 = 0$ and hard thresholding $\Theta_H(t;\lambda) = t1_{|t|\ge\lambda}$ with a properly chosen λ. This inspires us to apply an iterative procedure for screening: starting from $A^0 = 0$, repeat

1) $A^{k+1} \leftarrow A^k - \frac{1}{k_0^2}(\Sigma_{XX}A^k - \Sigma_{XY})$;
2) $\lambda^{k+1} \leftarrow (m+1)$th largest element of A^{k+1} in magnitude;
3) $A^{k+1} \leftarrow \Theta_H(A^{k+1};\lambda^{k+1})$;
4) $\mathcal{M}_\mu^{k+1} \leftarrow \{(i, j) : |a_{ij}^{k+1}| \neq 0, 1 \le i, j \le p\}$;

until $\mathcal{M}_\mu^k$ stops changing.

We call this screening procedure the *quantile thresholding iterative screening* (QTIS). As shown in Step 2 and Step 3, QTIS does not select variables using a fixed thresholding parameter λ. Instead, it uses a **dynamic** threshold to keep a fixed number m of nonzero elements at each iteration. The *quantile parameter* μ determines the number of variables to be selected. In comparison to SIS which ranks the connections based on $X^\top Y$, the iterative nature of QTIS lessens the greediness by repeatedly updating the importance of each candidate connection with a theoretical guarantee of convergence.

Theorem 2 *Given $k_0 > \|X\|_2$, for any $0 < \mu \leq 1$, the sequence of iterates $\{A^k\}$ generated by QTIS has the function value decreasing property that $l(A^{k+1}) \leq l(A^k)$, where $l(A) = \frac{1}{2}\|Y - XA\|_F^2$, and A^k satisfies $\|A^k\|_0 \leq m$, where $\|\cdot\|_0$ denotes the number of nonzero elements.*

See Appendix B for the detailed proof.

In practice, $\mathcal{M}_\mu$ usually stops changing after less than a hundred iterations. The number of unknowns is reduced from p^2 to $\lceil \mu pn \rceil$ effectively by a small amount of computation. Then, more involved and sophisticated estimation, for example, BISPS, can be performed to the reduced model. It is much faster than applying BISPS directly if $p^2 \gg n$. In addition, QTIS provides BISPS with a sparse pattern, which facilitates the fast computation of partial SVD.

To apply BISPS on $\mathcal{M}_\mu$, we use element-wise penalty parameters λ_{ij}'s and set

$$\lambda_{ij} = \infty \text{ if } (i,j) \notin \mathcal{M}_\mu.$$

This simple modification guarantees that only elements in $\mathcal{M}_\mu$ will be selected by BISPS.

3.5 Two-dimensional Selective Cross Validation for Tuning

The reparameterization of Berhu (9) separates the roles of ℓ_1 and ℓ_2 regularizations; each of them is associated with a regularization parameter, namely λ for ℓ_1 and η for ℓ_2. This provides important guidelines for parameter tuning. Based on our experience, the estimate is not very sensitive to η, so a full two-dimensional grid search is not necessary. Instead, we search along several one-dimensional solution paths including one η-path and three λ-paths:

- *Step 1*: Run the η-path ($\lambda = 0$). Do ridge regression with a grid of values for η, and choose the optimal η^* using AIC (Akaike, 1974).
- *Step 2*: Run 3 λ-paths with $\eta = 0.5\eta^*, 0.05\eta^*, 0.005\eta^*$ respectively. For each value of η, run BISPS with a grid of values for λ, and find the optimal one λ^o using the K-fold selective cross-validation (SCV) (She, 2012). This results in three λ^o's, one from each path. Choose the optimal one from them and let it be the optimal thresholding parameter λ^*. The pair (λ^*, η^*) is our final choice of the two parameters.

The SCV cross-validates different sparsity patterns instead of the regularization parameters. It is more computationally efficient than the plain cross validation since it runs the sparse algorithm only once and globally (instead of K times locally). To calculate the SCV error associated with λ, we first apply BISPS to the whole data set and obtain the solution $\hat{A}(\lambda)$. Then, for $k = 1, \cdots, K$, we apply ridge regression restricted to the variables that are picked by $nz(\hat{A}(\lambda))$ on the data without the kth data piece, and evaluate its validation error using the kth data piece. The sum of the K validation errors is defined as the SCV error. See She (2012) for more details.

4. Stationary Bootstrap (SB) Enhanced Network Learning

The $\mathbf{S}^2$ learning framework proposed in Section 3 is an effective technique to identify stationary and sparse network. Nevertheless, a "one-time" estimate, without any p-value or confidence interval, provides only limited guidance in identifying the true network topology. In fact, whatever inference method is used, there will be uncertainty underlying the variable selection procedure. It would be greatly helpful if one could provide some kind of uncertainty measure for such an estimate. In our case, we would like to find a certain confidence measure for the estimated topology. This can be done by assigning a probability for the existence of each connection. Hence, we use bootstrap (Efron, 1979). In this section, we propose the *stationary bootstrap* enhanced BISPS (SB-BISPS) which provides a confidence level about whether a connection exists in the network by measuring the frequency with which it is chosen by the BISPS algorithm.

4.1 The SB-BISPS Framework

The SB-BISPS procedure completes the BISPS (or QTIS+BISPS) algorithm with a stationary bootstrap resampling step. The SB-BISPS is described as follows.

- *Step 1*: Run BISPS over the original data set $\mathcal{X}$. Record the pattern of $\hat{A}$, which is a $p \times p$ binary matrix $\Phi = [\phi_{ij}]_{1 \le i,j \le p}$ defined as:

$$\phi_{ij} = \begin{cases} 1 & \text{if } \hat{a}_{ij} \neq 0 \\ 0 & \text{if } \hat{a}_{ij} = 0. \end{cases}$$

- *Step 2*: Draw B **stationary** bootstrap samples from $\mathcal{X}$. Repeat Step 1 for each sample. Record Φ_j^* for the jth sample.

- *Step 3*: Compute the matrix $F = [f_{ij}]_{1 \le i,j \le p}$ of *connection occurring frequency* (COF) by adding up all the patterns Φ_j^*'s and normalizing the result by B:

$$F = \frac{1}{B} \sum_{j=1}^{B} \Phi_j^*.$$

Given a sufficiently large B, f_{ij} is a good approximation of the probability for BISPS to select connection a_{ij}, which serves as a measure of how confident we are with the existence of this connection. For example, if $f_{ij} = 80\%$, it means that in 80% of the bootstrap samples, a connection from node i to node j is detected. So we can say the probability for the existence of this connection is (approximately) 80%. We can use a cutoff value f^* to threshold the COF matrix, and choose only the connections with $f_{ij} \ge f^*$ for further analysis. This renders us a sparse topology that shows the most significant connections within the network.

4.2 Stationary Bootstrap

There are different resampling schemes to draw bootstrap samples from the original data in Step 2. If the observations are independent and identically distributed, we can resample the data randomly with replacement. When the observations are time series, the problem is more complicated, since the observations are largely dependent on each other, and we would like to preserve the specific

dependency structure in bootstrapping. Techniques such as resampling blocks of consecutive observations or resampling "blocks of blocks" can be used (Kunsch, 1989). The basic idea is that, despite the dependence of individual observations, blocks of observations can be approximately independent with each other given a proper block size l.

When a time series is stationary, it is natural to maintain this property in the bootstrap samples. The stationary bootstrap (Politis and Romano, 1994) is a method with this property. It is based on resampling blocks of random lengths, where the length of each block follows a geometric distribution with mean $1/\gamma$. We apply a simple approach to conduct such resampling. Given that x_i^* is chosen to be the Jth observation x_J in the original time series, we choose x_{i+1}^* based on the following rule:

$$x_{i+1}^* = \begin{cases} x_{J+1} \text{ with probability } 1-\gamma \\ \text{picked randomly from } x_1, \cdots, x_n \text{ with probability } \gamma. \end{cases}$$

Similar with block bootstrap, where the block size l has to be determined, the value of γ should be chosen properly. Fortunately, the sensitivity of γ in stationary bootstrap is less than that of l in block bootstrap.

5. Experiments

In this section, we present the experimental results on synthetic data and demonstrate the effectiveness of the proposed $\mathbf{S}^2$ network learning framework.

5.1 Performance Measures and Experiment Settings

To examine the performance of the proposed methods, we define the following measures.

- Stationarity violation percentage (P_v): In N repeated experiments, if there are N_v experiments in which the estimate $\hat{A}$ violates the stationarity condition (4), then the stationarity violation percentage is defined as $P_v = N_v/N$.
- Miss rate (P_m): If $a_{ij} \neq 0, \hat{a}_{ij} = 0$, we say there is a miss. Denote C_m as the total number of misses and C_{nz} as the number of nonzero entries in A. The miss rate is defined as $P_m = C_m/C_{nz}$.
- False alarm rate (P_f): If $a_{ij} = 0, \hat{a}_{ij} \neq 0$, we say there is a false alarm. Denote C_f as the total number of false alarms and C_z as the number of zero entries in A. The false alarm rate is defined as $P_f = C_f/C_z$.
- Testing error (TE): The testing error is defined as $TE = \frac{1}{n_t}\|Y^t - X^t\hat{A}\|_F^2$, where Y^t and X^t are testing data, and n_t is their length. For time series, the testing data are collected right after the training data.
- Computation time: The averaged running time of an algorithm. All the algorithms are run in MATLAB7.11.0 on a PC with 4GB memory.

Particularly, to evaluate the prediction/forecasting performances of the algorithms, we adopt the so-called rolling MSE, a conventional measure in econometrics (Stock and Watson, 2012). Suppose we have T observations: $x_1, \cdots, x_T$. Let the rolling window size be W and the horizon be h. Standing at time t, we use the most recent W observations to estimate A, denoting the estimate as $\hat{A}_t$. Then

we use this estimate to forecast x_{t+h}, denoting the forecast as $\hat{x}_{t+h}$ and the forecasting error as $e_t^h = \|x_{t+h} - \hat{x}_{t+h}\|_2^2$. This process is repeated for $t = W, \cdots, W+N-1$ as we shift the window. N is the number of window shifting that satisfies $1 \le N \le T - W - h - 1$. Then the rolling MSE for horizon h is defined as $MSE_{rolling}^h = \frac{1}{N}\sum_{t=W}^{W+N-1} e_t^h$. When using $\hat{A}_t$ to forecast x_{t+h}, we should do pseudo out-of-sample forecasting. That is, we assume the observations after t are not available and consequently we need do h-step-ahead forecast. For our model (1), this should be done as: $\hat{x}_{t+h} = \hat{A}_t^\mathsf{T} \hat{x}_{t+h-1}$ for $h \ge 1$, where $\hat{x}_t \stackrel{\Delta}{=} x_t$ when $h = 1$. The testing error defined above corresponds to the rolling MSE for $h = 1$.

We generate the $p \times p$ transition matrix A with both sparsity and stationarity properties. First, the topology is generated from a directed random graph $G(p, \xi)$, where the edge from one node to another node occurs independently with probability ξ. Then the strength of the edges is generated independently from a Gaussian distribution. This process is repeated until we obtain a matrix A that has a desired spectral radius $0.9 < \rho(A) < 1$. We set $\xi = 10/p, \Sigma_\varepsilon = \sigma^2 I, \sigma^2 = 10$.

The regularization parameters are chosen by SCV as described in Section 3.5. For a λ-path, we use a grid of 100 values for λ, which is picked from the interval $[0, \|A^0 + X^\mathsf{T}Y - X^\mathsf{T}XA^0\|_{\max}]$. The initial estimate is simply set as $A^0 = 0$. For an η-path, we use a grid of 76 values for η, which is picked from the interval $[2^{-10}, 2^5]$. The number of folds for SCV is set to be $K = 5$. All the statistics collected are averaged over $N = 100$ times of window shifting. The length of testing data $n_t = 200$.

5.2 Performance of BISPS

We compare the performance of BISPS with Lasso, eNet and Berhu—we use the penalty name to denote the corresponding PML estimation. The number of observations is $n = 80$. Table 1 shows the experiment results for different network sizes, namely $p = 100, 200, 300$. Recall that for a network with size p, the number of unknown parameters is p^2. For example, for the network with 300 nodes, the number of parameters to be estimated is 9×10^4, which is extremely high compared with the number of observations 80.

Among the three penalties, the Lasso solution gives higher miss rates. This is because when some predictors are correlated, Lasso tends to choose only a part, or even none, of them. As a result, Lasso sometimes "over-shrinks" the estimate. The eNet and Berhu, in such cases, tend to include all the correlated predictors, thanks to the ℓ_2 part in the penalties. However, the sparsity of the eNet solution is affected by the ℓ_2 regularization, so it gives high false alarm rates. On the other hand, Berhu has improved eNet to some extend by enforcing the ℓ_2 regularization only to large coefficients. As a result, Berhu achieves the smallest testing errors ($h = 1$) among the three penalties.

As shown by P_v, no matter what penalty is used, it is possible for PML to give a nonstationary estimate, whereas the proposed $\mathbf{S}^2$ learning and BISPS can guarantee the stationarity property of $\hat{A}$. This indicates that adding the stationarity constraint into the sparsity pursuit does effectively prevent the estimate from becoming nonstationary. Meanwhile, the $\mathbf{S}^2$ estimate can achieve a comparable estimation and detection accuracy with the PML estimate. Table 1 gives the rolling MSEs for different horizons h to illustrate both the short term and long term forecasting performance. For PML estimates, the rolling MSEs grow *explosively* with h due to the existence of nonstationary estimates, while those of BISPS accumulate much more slowly.

To further illustrate the disadvantages of a nonstationary estimate, we find one run where Lasso gives a nonstationary estimate $\hat{A}_{Lasso}$. Starting from a time point t, we generate $\hat{x}_{t+h}(\hat{A})$

p	method	P_v	$(P_m, P_f)(\%)$	$h=1$	$h=64$	$h=128$
100	Lasso	5%	(13.6, 22.3)	23.4	1058.3	7798.9
	eNet	5%	(12.3, 25.9)	23.5	1238.1	10842.8
	Berhu	5%	(12.3, 24.9)	23.0	1329.4	12219.6
	BISPS	**0%**	(12.4, 24.8)	23.1	195.9	**209.2**
200	Lasso	3%	(15.8, 29.3)	41.8	332.8	10998.7
	eNet	2%	(14.5, 26.9)	36.5	207.8	405.3
	Berhu	2%	(14.2, 24.6)	32.2	201.4	391.5
	BISPS	**0%**	(14.2, 24.4)	32.1	131.9	**200.5**
300	Lasso	2%	(18.1, 14.9)	19.9	61.6	111.2
	eNet	3%	(13.5, 26.1)	20.4	65.5	123.2
	Berhu	3%	(13.4, 22.9)	18.8	66.5	121.3
	BISPS	**0%**	(13.7, 22.1)	19.3	63.3	**65.5**

Table 1: Performance comparison of BISPS with Lasso, eNet and Berhu. $n = 80$.

for $h = 1, \cdots, 100$ using $\hat{A}_{Lasso}$ and $\hat{A}_{BISPS}$ respectively and compare them with x_{t+h} observed from the true model. The results are plotted in Figure 5. We can easily see that $\hat{x}(\hat{A}_{BISPS})$ gives a reasonable imitation of the true system. The nonstationary estimate $\hat{x}(\hat{A}_{Lasso})$, however, *blows up* quickly and behaves completely differently from the true model. This indicates that ensuring a stationary estimate is indeed crucial.

5.3 Performance of QTIS

To examine the performance of QTIS for connection screening, we first compare it with the sure independence screening (SIS) (Fan and Lv, 2008) by examining their ability to include all the true connections, which can be measured by the miss rate. Simulation is done for networks with different sizes, namely $p = 300, 400, 500$. The sample size $n = 80$. Independence screening methods, including SIS, are very popular in ultra-high dimensional problems for dimension reduction and variable selection. However, our finding is that such methods can perform very poorly for network learning. As shown in Figure 6, it is possible for SIS to miss even more than half of the true connections. One possible reason is that, because of the evolving processes, correlation exits ubiquitously in dynamical networks. As a result, independence screening is not appropriate for network learning. On the other hand, the proposed QTIS algorithm considers the correlation issue and thus can obtain much smaller miss rates than SIS. Also, its performance is more robust to the choice of the quantile parameter μ.

We then run BISPS with and without QTIS and check the difference of their performances. Denote "QTIS+BISPS" as the procedure that first applies QTIS to screen the connections and then applies BISPS to the reduced model. Networks with sizes up to $p = 800$ are considered (the number of unknown parameters is $p^2 = 640,000$). The sample size $n = 80$. The quantile parameter $\mu = 0.8$. Figure 7 compares the performance of QTIS+BISPS and BISPS. When p/n ratio is large, adding QTIS not only improves the estimation and identification accuracy, but also saves up to **80%** of the computation time. As the p/n ratio becomes larger, the improvement becomes more remarkable. Therefore, QTIS is a helpful tool to facilitate BISPS for ultra-high dimensional network learning.

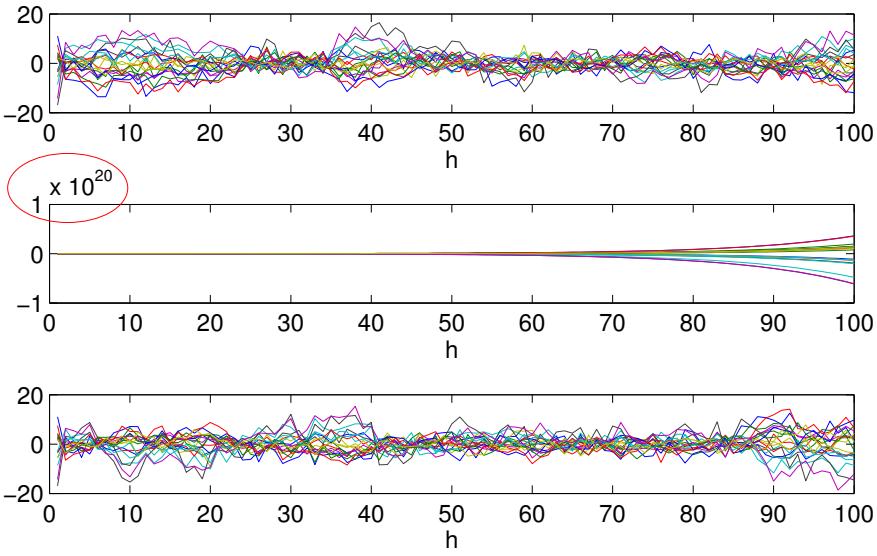


Figure 5: Comparison of BISPS and Lasso in terms of forecasting performance. Top: sample from the true model; Middle: forecast from the nonstationary estimate $\hat{A}_{Lasso}$; Bottom: forecast from the stationary estimate $\hat{A}_{BISPS}$.

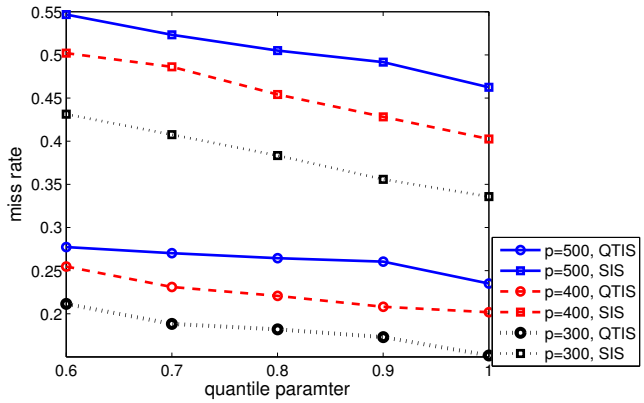
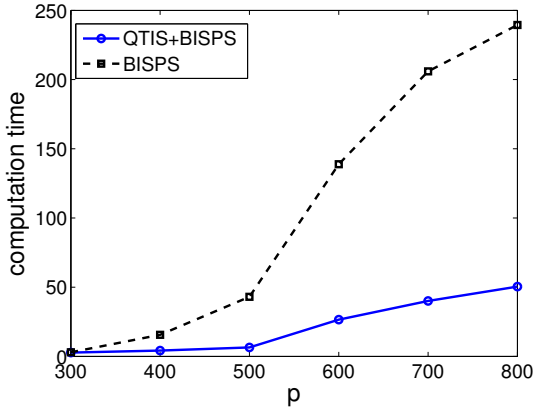


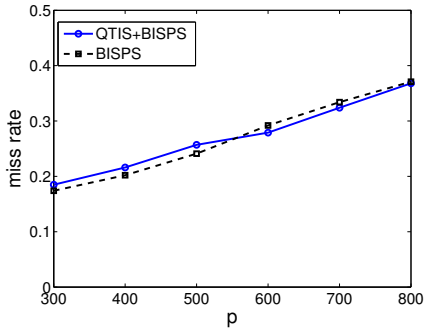
Figure 6: Miss rate comparison for QTIS and SIS. $n = 80$.

6. Application to U.S. Macroeconomic Data

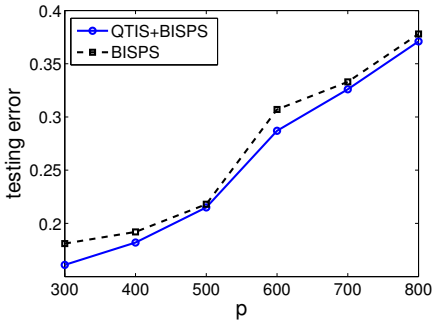
We apply the proposed learning framework to the U.S. macroeconomic data. The data set consists of quarterly observations on 108 macroeconomic variables from 1960:I to 2008:IV, which belong to 12 categories. A complete description of the data can be found at "`http://www.princeton.edu/~mwatson/wp.html`". There are in total 109 macroeconomic variables from 13 categories. We remove Category 13 (consumer expect) since it has only one variable while we are interested



(a) computation time



(b) miss rate



(c) testing error

Figure 7: Performance of QTIS+BISPS, compared with applying BISPS to a full model. $n = 80$.

Category	Lasso	**BISPS**	Category	Lasso	**BISPS**
1. GDP	0.589	0.445	7. Prices	1.971	1.874
2. IP	0.846	0.576	8. Wages	0.552	0.207
3. Employment	0.936	0.711	9. Interest rate	1.443	0.738
4. Unempl. rate	0.289	0.165	10. Money	0.114	0.065
5. Housing	0.071	0.033	11. Exchange rates	0.370	0.107
6. Inventories	0.506	0.217	12. Stock prices	0.254	0.100

Table 2: Normalized Rolling MSE of Lasso and BISPS for each category.

in multivariate time series. The data have been preprocessed so that each time series is a stationary process. We use the $\mathbf{S}^2$ network model (1) to detect Granger causal relations between these macroeconomic indices.

h	1	2	4	8	16	32
Lasso	0.017	0.021	0.029	0.365	329.9	3.1×10^8
BISPS	0.017	0.018	0.019	0.020	**0.020**	**0.018**

Table 3: Rolling MSE of Lasso and BISPS for different horizons.

6.1 Comparison of Rolling MSE

We first study the data set by category, considering that multiple time series explain the interactions of the indices in each category. To each of the 12 categories, we apply Lasso and BISPS respectively with the horizon $h = 1$ and the rolling window size $W = 0.8 \times p$, where p is the number of time series (network size). Table 2 shows the rolling MSEs of the Lasso and BISPS, normalized by that of the AR(4) model, which is a conventional benchmark of macroeconomic forecasting.

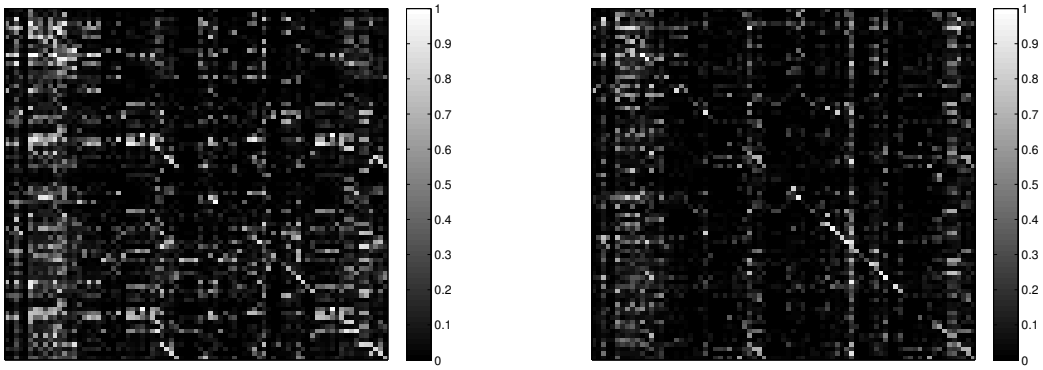
Compared with the AR(4) model, both Lasso and BISPS, based on a VAR model, have attained much smaller forecasting errors, except for Category 7 (prices). Therefore, by introducing the Granger causal interactions between different indices, we can build a multivariate network model that is more accurate than the univariate AR model in capturing the evolution of the U.S. macroeconomics, given the same amount of observations. The exception of Category 7 may be due to the higher lag order used in the univariate AR model.

Moreover, we note that BISPS gives smaller forecasting errors than Lasso for all the 12 categories of macroeconomic time series. It indicates that adopting a fusion of ℓ_1 and ℓ_2 penalties and imposing the stationarity constraint can capture the network dynamics more accurately and achieve a stronger capability of forecasting. To further support this conclusion, we apply Lasso and BISPS respectively to all the 108 variables with $W = 0.8 \times p$ and different horizons h. The rolling MSEs for $h = 1, 2, 4, 8, 16, 32$ is recorded in Table 3. As the horizon increases, the rolling MSE of Lasso grows exponentially, which clearly indicates that some estimates of the Lasso are nonstationary and thus fail to forecast for large horizons. On the other hand, the rolling MSE of BISPS stays stable for different horizons. This phenomenon is similar to what is shown in Figure 5. They have illustrated the fundamental difference of the $\mathbf{S}^2$ learning from the plain PML estimation.

6.2 Bootstrap Analysis

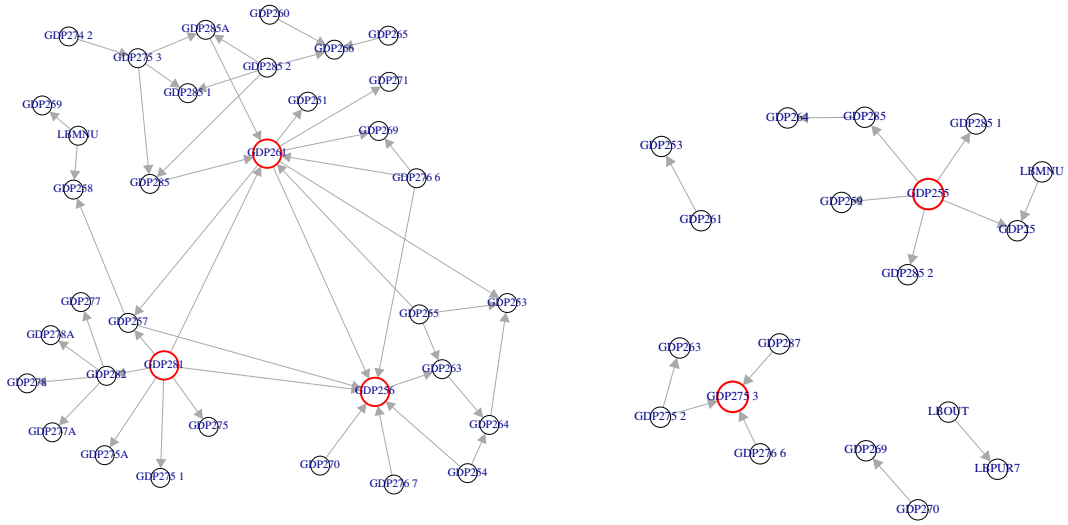
In this experiment, we apply the SB-BISPS to the macroeconomic data before and after the "Great Moderation" (Davis and Kahn, 2008) and analyze the changes in their Granger causal connections. As the economic structure of U.S. has gone through huge changes in the Great Moderation in mid-1980, we expect to see significantly different causality networks before and after mid-1980. Hence, we divide the time series into two periods, the pre-Great Moderation period and the post-Great Moderation period, and apply SB-BISPS separately to the two periods.

For the pre-Great Moderation period, we use the data from 1960:I to 1979:IV as training set (80 observations); for the post-Great Moderation period, we use the data from 1985:I to 2004:IV (80 observations). The stationary bootstrap samples are obtained using the R function *tsboot* (Dalgaard, 2008) with default parameter values. The number of stationary bootstrap samples is set to be $B = 100$. Figure 8 shows the COF (connection occurring frequency) matrices given by SB-BISPS for the pre-Great Moderation and the post-Great Moderation periods. We notice that the COF matrix of the pre-Great Moderation period has a higher energy level than that of the post-Great Moderation



(a) pre-Great Moderation (b) post-Great Moderation

Figure 8: COFs of the pre-Great Moderation period and the post-Great Moderation period.



(a) pre-Great Moderation (b) post-Great Moderation

Figure 9: Topologies of the macroeconomic network in the pre-Great Moderation period and the post-Great Moderation period. $f^* = 80\%$. Self-loops are not shown.

period. This indicates that the nodes are more actively interacting with each other in the pre-Great Moderation period than in the post-Great Moderation period, which effectively reflects the reduction in volatility of the business cycle fluctuations since the Great Moderation.

To illustrate the idea more clearly, we set the cutoff value for COF to be $f^* = 80\%$ and identify the most significant connections. The topologies obtained for the pre-Great Moderation period and the post-Great Moderation period are shown in Figure 9. Isolated nodes are removed. In the pre-Great Moderation period, the macro variables actively interact and form a complex dynamical

network. There are three prominent variables, namely GDP281 (durable goods index), GDP256 and GDP261 (gross private domestic investment indices), which act like "hub" variables. They interact not only with many non-hub variables but also with each other. Therefore, there are no independent clusters. After the Great Moderation, on the other hand, the interactions have been remarkably reduced and most of the variables seem only self-regulated. This makes it easier for the network to stay stable. There are two hub variables, GDP255 (real personal consumption expenditure) and GDP275-3 (energy goods price index). The increasing importance of these two variables agrees with the observation that environmental regulations and energy policies have begun to influence the economic growth since the Great Moderation period (Jorgenson and Wilcoxen, 1990; Halkos and Tzeremes, 2011).

7. Conclusion

We have proposed the stationary-sparse ($\mathbf{S}^2$) learning of causality networks described in Granger's sense. Distinguished from the existing works, we explicitly incorporated the stationarity concern in a possibly ultra-high dimensional scenario and provided a probabilistic measure for the occurrence of any causal connection. We added a relaxed stationarity constraint in the penalized maximum likelihood estimation and proposed the BISPS algorithm which is easy to implement and computationally efficient. We must point out that although the algorithm is designed for the Berhu penalty, the framework extends to any convex penalties and their coupled thresholding rules. In network modeling, the number of unknown variables p^2 is often much larger than the number of observations n, which confronts us with an ultra-high dimensional problem. Therefore, we implanted the quantile thresholding iterative screening (QTIS) into the BISPS algorithm to improve scalability and computational efficiency. Furthermore, the stationary bootstrap enhanced BISPS (SB-BISPS) was proposed to provide a confidence measure for each possible connection in the network. The method has been successfully applied to the U.S. macroeconomic data, which leads to some interesting discoveries.

Our current work assumes multivariate Gaussian noise and focuses on learning the transition matrix. We will pursue the network learning in more general settings in the future. One particular problem of interest is to jointly capture the structure of the transition matrix and the concentration matrix, which may provide more comprehensive descriptions of the network. Also, we will consider the situations where the noise follows other distributions other than Gaussian, for example, the heavy-tail distribution. Finally, we will proceed to nonlinear time series models for network identification to handle more complex network data.

Acknowledgments

We would like to thank the editor and the anonymous referees for their careful comments and useful suggestions that significantly improve the quality of the paper. This work is partially supported by NSF under grants CCF-1117012 and CCF-1116447.

Appendix A. Proof of Theorem 1

We begin the proof by introducing some lemmas. Throughout the proof, we assume $\tau \geq 0, \lambda \geq 0, \eta \geq 0$, and $\lambda\eta \neq 0$.

Lemma 3 *Given the Berhu penalty* (9)*, the minimization problem*

$$\min_B \frac{1}{2}\|B-\Xi\|_F^2 + P_{\mathcal{B}}(B;\lambda,\eta) \tag{12}$$

has a unique optimal solution given by $\hat{B} = \Theta_{\mathcal{B}}(\Xi;\lambda,\eta)$*, where* $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ *is the Berhu thresholding rule defined as* (10)*.*

Proof It is easy to verify that $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is an odd, nondecreasing, shrinkage function that satisfies the definition of a thresholding rule given in She (2009). Following the construction procedure

$$P(t;\lambda,\eta) = \int_0^{|t|} (\sup\{s : \Theta(s;\lambda,\eta) \le u\} - u)du, \tag{13}$$

the Berhu penalty $P_{\mathcal{B}}(\cdot;\lambda,\eta)$ can be constructed from $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$. So $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is the coupled thresholding rule for $P_{\mathcal{B}}(\cdot;\lambda,\eta)$. By Lemma 1 in She (2012), $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is the global minimizer of (12). ■

Lemma 4 *The Berhu thresholding operator* $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ *is nonexpansive, that is,* $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)| \le |t-\tilde{t}|$ *for any* $t,\tilde{t} \in \mathbb{R}$*.*

The conclusion directly extends to the multivariate case: $\|\Theta_{\mathcal{B}}(A;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{A};\lambda,\eta)\| \le \|A-\tilde{A}\|$ for any $A,\tilde{A} \in \mathbb{R}^{p\times q}$.

Proof It is sufficient to show that the univariate Berhu thresholding operator $\Theta_{\mathcal{B}}$ is nonexpansive. Define $\Delta = |t-\tilde{t}|^2 - |\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)|^2$ and $a = |t|, b = |\tilde{t}|$.

a) Suppose $a \le \lambda, b \le \lambda$. Then $\Theta_{\mathcal{B}}(t;\lambda,\eta) = \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta) = 0$. So $\Delta = |t-\tilde{t}|^2 \ge 0$.

b) Suppose $a \le \lambda, \lambda < b \le \lambda + \lambda/\eta$. Then $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)|^2 = |\tilde{t} - \lambda sng(\tilde{t})|^2 = b^2 + \lambda^2 - 2\lambda b$. So $\Delta = a^2 + b^2 - 2ab - (b^2 + \lambda^2 - 2\lambda b) = (\lambda - a)(2b - a - \lambda) \ge 0$.

c) Suppose $a \le \lambda, b \ge \lambda + \lambda/\eta$. Then $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)|^2 = |\frac{\tilde{t}}{1+\eta}|^2$. So $\Delta = a^2 + b^2 - 2ab - \frac{b^2}{(1+\eta^2)} = a^2 + [\frac{\eta^2+2\eta}{(1+\eta)^2}b - 2a]b \ge a^2 + [\frac{\eta^2+2\eta}{(1+\eta)^2}(\lambda + \lambda/\eta) - 2a]\lambda = (a-\lambda)^2 + \frac{1}{1+\eta}\lambda^2 \ge 0$.

d) Suppose $\lambda < a \le \lambda + \lambda/\eta, \lambda < b \le \lambda + \lambda/\eta$. Then $\Delta = 2\lambda(1 - \mathrm{sgn}(t\tilde{t}))(a + b - \lambda) \ge 0$.

e) Suppose $\lambda < a \le \lambda + \lambda/\eta, b \ge \lambda + \lambda/\eta$. Then $\Delta = |t-\tilde{t}|^2 - |t - \lambda\mathrm{sgn}(t) - \frac{\tilde{t}}{\eta+1}|^2 = \frac{\eta(\eta+2)}{(\eta+1)^2}b^2 - 2b\frac{\eta a+\lambda}{\eta+1}\mathrm{sgn}(t\tilde{t}) + \lambda(2a-\lambda) = b[\frac{\eta(\eta+2)}{(\eta+1)^2}b - 2\frac{\eta a+\lambda}{\eta+1}\mathrm{sgn}(t\tilde{t})] + \lambda(2a-\lambda) \ge (\lambda+\lambda/\eta)[\frac{\eta(\eta+2)}{(\eta+1)^2}(\lambda+\lambda/\eta) - 2\frac{\eta a+\lambda}{\eta+1}\mathrm{sgn}(t\tilde{t})] + \lambda(2a-\lambda) = \frac{\lambda}{\eta}[\eta\lambda + 2\lambda - 2(\eta a + \lambda)\mathrm{sgn}(t\tilde{t})] + \lambda(2a-\lambda) \ge 0$.

f) Suppose $a \ge \lambda + \lambda/\eta, b \ge \lambda + \lambda/\eta$. Then $\Delta = |t-\tilde{t}|^2 - |\frac{t}{1+\eta} - \frac{\tilde{t}}{1+\eta}|^2 = \frac{\eta^2+2\eta}{(1+\eta)^2}|t-\tilde{t}|^2 \ge 0$.

Therefore, $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)| \le |t-\tilde{t}|$ for any $t,\tilde{t} \in \mathbb{R}$. So the Berhu thresholding operator is nonexpansive. ■

Lemma 5 *Let the SVD of* B *be* $B = USV^{\mathsf{T}}$*, where* $S = diag(\nu_1,\nu_2,\cdots,\nu_p)$ *with* $\nu_1,\nu_2,\cdots,\nu_p$ *being the singular values. Then* $\Pi(B;\tau)$ *defined by*

$$\Pi(B;\tau) = U diag(\min(\nu_1,\tau),\cdots,\min(\nu_p,\tau))V^{\mathsf{T}}$$

gives the projection of B *into the convex set* $\{B : \|B\|_2 \le \tau\}$*.*

Proof Let C be the projection of B into the convex set $\{B : \|B\|_2 \leq \tau\}$. Then C can be solved by

$$\min_C \|B-C\|_F^2,$$
$$\text{s.t. } \|C\|_2 \leq \tau.$$

To prove the lemma, we introduce von Neumann's trace inequality (von Neumann, 1937), which states that for any $p \times p$ matrices A and B with singular values $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_p$ and $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_p$ respectively,

$$|\text{tr}\{AB\}| \leq \sum_{i=1}^{p} \alpha_i \beta_i, \tag{14}$$

where equality holds if and only if it is possible to find unitary matrices U and V that simultaneously singular value decompose A and B.

Let $B = U_0 S_0 V_0^\mathsf{T}$ and $C = USV^\mathsf{T}$ be the singular value decompositions of B and C respectively, where $S_0 = \text{diag}(\nu_{0,1}, \nu_{0,2}, \cdots, \nu_{0,p})$ and $S = \text{diag}(\nu_1, \nu_2, \cdots, \nu_p)$ with $\nu_{0,1} \geq \nu_{0,2} \geq \cdots \geq \nu_{0,p}$ and $\nu_1 \geq \nu_2 \geq \cdots \geq \nu_p$. Then,

$$\begin{aligned}\|B-C\|_F^2 &= \|B\|_F^2 + \|C\|_F^2 + 2\text{tr}\{B^\mathsf{T}C\} \\ &\geq \|S_0\|_F^2 + \|S\|_F^2 + 2\text{tr}\{S_0 S\} \\ &= \sum_{i=1}^{p} (\nu_{0,i} - \nu_i)^2.\end{aligned}$$

Equality holds if and only if $U = U_0$ and $V = V_0$. Optimality is achieved at $\nu_i = \min(\nu_{0,i}, \tau), i = 1, \cdots, p$. The proof is complete. ■

Lemma 6 *The projection operator $\Pi(\cdot;\tau)$ defined in Lemma 5 is nonexpansive, that is, $\|\Pi(A) - \Pi(\tilde{A})\|_F \leq \|A - \tilde{A}\|_F$ for any $A, \tilde{A} \in \mathbb{R}^{p\times p}$.*

Proof For simplicity, we denote $\Pi(\cdot;\tau)$ as $\Pi(\cdot)$. Let the SVDs for $p \times p$ matrices A and $\tilde{A}$ be $A = UDV^\mathsf{T}$ and $\tilde{A} = \tilde{U}\tilde{D}\tilde{V}^\mathsf{T}$ respectively. Define $\Delta = \|A - \tilde{A}\|_F^2 - \|\Pi(A) - \Pi(\tilde{A})\|_F^2$. Then,

$$\begin{aligned}\Delta &= \|UDV^\mathsf{T} - \tilde{U}\tilde{D}\tilde{V}^\mathsf{T}\|_F^2 - \|U\Pi(D)V^\mathsf{T} - \tilde{U}\Pi(\tilde{D})\tilde{V}^\mathsf{T}\|_F^2 \\ &= \|D\|_F^2 + \|\tilde{D}\|_F^2 - \|\Pi(D)\|_F^2 - \|\Pi(\tilde{D})\|_F^2 - 2\text{tr}\{VDU^\mathsf{T}\tilde{U}\tilde{D}\tilde{V}^\mathsf{T}\} + 2\text{tr}\{V\Pi(D)U^\mathsf{T}\tilde{U}\Pi(\tilde{D})\tilde{V}^\mathsf{T}\}.\end{aligned}$$

Applying von Neumann's trace inequality (14) again, we have

$$\begin{aligned}&-2\text{tr}\{VDU^\mathsf{T}\tilde{U}\tilde{D}\tilde{V}^\mathsf{T}\} + 2\text{tr}\{V\Pi(D)U^\mathsf{T}\tilde{U}\Pi(\tilde{D})\tilde{V}^\mathsf{T}\} \\ &= -2\text{tr}\{V(D-\Pi(D))U^\mathsf{T}\tilde{U}\tilde{D}\tilde{V}^\mathsf{T} + V\Pi(D)U^\mathsf{T}\tilde{U}(\tilde{D}-\Pi(\tilde{D}))\tilde{V}^\mathsf{T}\} \\ &\geq -2\{\sum_{i=1}^{p}(d_i - \Pi(d_i))\tilde{d_i} + \sum_{i=1}^{p}(\tilde{d_i} - \Pi(\tilde{d_i}))\Pi(d_i)\}.\end{aligned}$$

Therefore,

$$\Delta \geq \sum_{i=1}^{p}\{(d_i - \tilde{d_i})^2 - (\Pi(d_i) - \Pi(\tilde{d_i}))^2\}.$$

It is easy to verify that $(d_i - \tilde{d}_i)^2 - (\Pi(d_i) - \Pi(\tilde{d}_i))^2 \geq 0, i = 1, \cdots, p$. So $\Delta \geq 0$. The projection Π is a nonexpansive mapping. ■

Lemma 7 *Let $P^0 = Q^0 = 0$. The sequence $\{B^j\}$ of iterative procedure*

$$\begin{aligned} C^j &= \Theta_{\mathcal{B}}(B^j + P^j; \lambda, \eta), \\ P^{j+1} &= B^j + P^j - C^j, \\ B^{j+1} &= \Pi(C^j + Q^j; \tau), \\ Q^{j+1} &= C^j + Q^j - B^{j+1} \end{aligned} \tag{15}$$

converges to a globally optimal solution to

$$\begin{aligned} &\min_B \frac{1}{2}\|B - B^0\|_2^2 + P_{\mathcal{B}}(B; \lambda, \eta), \\ &\qquad s.t.\ \|B\|_2 \leq \tau. \end{aligned} \tag{16}$$

Procedure (15) is designed for a penalized minimization problem with a convex constraint based on Dykstra's projection algorithm (Dykstra, 1983; Boyle and Dykstra, 1986).

Proof First, we rewrite the problem as

$$\min_B \frac{1}{2}\|B - B^0\|_2^2 + f(B) + g(B), \tag{17}$$

where $f(B) = P_{\mathcal{B}}(B; \lambda, \eta)$ and $g(B) = \mathbb{1}_{\|B\|_2 \leq \tau}$ is an indicator function for $\|B\|_2 \leq \tau$, defined as

$$\mathbb{1}_{\|B\|_2 \leq \tau} = \begin{cases} 0 & \text{if } \|B\|_2 \leq \tau \\ +\infty & \text{otherwise.} \end{cases}$$

It is easy to show that $g(B)$ is a proper lower semicontinuous convex function, $f(B)$ is a proper continuous (hence lower semicontinuous) convex function (Rockafellar, 1970) and they satisfy

$$\mathrm{dom} f \cap \mathrm{dom} g \neq \emptyset.$$

Lemma 3 and Lemma 5 imply that $\Theta_{\mathcal{B}}(\cdot; \lambda, \eta)$ and $\Pi(\cdot; \tau)$ are the proximity operators (Moreau, 1962) of $f(B)$ and $g(B)$ respectively:

$$\mathrm{prox}_f B = \Theta_{\mathcal{B}}(B; \lambda, \eta) \text{ and } \mathrm{prox}_g B = \Pi(B; \tau).$$

Therefore, by Theorem 3.2 and Theorem 3.3 in Bauschke and Combettes (2008), it holds that

$$B^j \to \mathrm{prox}_{f+g} B^0.$$

Hence, the sequence $\{B^j\}$ converges to a globally optimal solution to problem (17). ■

Now we can establish Theorem 1. Recall that Algorithm 1 is to solve

$$\min_A f(A;\lambda,\eta) = \frac{1}{2}\|Y - XA\|_F^2 + P_{\mathcal{B}}(A;\lambda,\eta),$$
$$\text{s.t. } \|A\|_2 \le 1.$$

We use Opial's conditions (Opial, 1967) to prove the convergence of $\{A^k\}$. To be specific, we show that the iteration of Step 1 to Step 3 in Algorithm 1 is a nonexpansive asymptotically regular mapping with a nonempty set of fixed points.

Lemma 8 *For the sequence $\{A^k\}$ generated by Algorithm 1,*

$$f(A^k;\lambda,\eta) - f(A^{k+1};\lambda,\eta) \ge \frac{1}{2}(k_0^2 - \|X\|_2^2)\|A^{k+1} - A^k\|_F^2.$$

Proof First define

$$g(A,B;\lambda,\eta) = \frac{1}{2}\|Y - XB\|_F^2 + P_{\mathcal{B}}(B;\lambda,\eta) + \frac{1}{2}\text{tr}\{(B-A)^\top(k_0^2 I - X^\top X)(B-A)\}.$$

Given A, minimizing g over B is equivalent to

$$\min_B \frac{1}{2}\|B - A - \frac{1}{k_0^2}(X^\top Y - X^\top XA)\|_F^2 + P_{\mathcal{B}}(B;\lambda/k_0^2,\eta/k_0^2)$$
$$\text{s.t. } \|B\|_2 \le 1.$$

We can obtain its globally optimal solution by performing the iterative procedure (15) in Lemma 7, substituting $\tau \leftarrow 1, B^0 \leftarrow A + \frac{1}{k_0^2}(X^\top Y - X^\top XA), \lambda \leftarrow \lambda/k_0^2, \eta \leftarrow \eta/k_0^2$. Therefore, we have

$$f(A^k;\lambda,\eta) = g(A^k,A^k;\lambda,\eta) \ge g(A^k,A^{k+1};\lambda,\eta) = f(A^{k+1};\lambda,\eta) + \frac{1}{2}(k_0^2 - \|X\|_2^2)\|A^{k+1} - A^k\|_F^2.$$

The proof is complete. ■

Given $k_0 > \|X\|_2$, Lemma 8 implies that the sequence $\{A^k\}$ is asymptotically regular (Browder and Petryshyn, 1966).

Lemma 9 *The sequence $\{A^k\}$ generated by the iteration of Algorithm 1 is uniformly bounded.*

Proof First, based on Lemma 8 we have

$$P_{\mathcal{B}}(A^k;\lambda,\eta) \le f(A^k;\lambda,\eta) \le f(A^0;\lambda,\eta) \triangleq C.$$

This implies that $P_{\mathcal{B}}(a_{ij}^k;\lambda,\eta) \le C, \forall 0 \le i,j \le p$.

If $|a_{ij}^k| \le \lambda/\eta$, we have $\lambda|a_{ij}^k| \le C$, which implies $(a_{ij}^k)^2 \le \max(\lambda^2/\eta^2, C^2/\lambda^2)$. If $|a_{ij}^k| > \lambda/\eta$, we have $\frac{\eta^2 t^2 + \lambda^2}{2\eta} \le C$, which implies $(a_{ij}^k)^2 \le \frac{2\eta C - \lambda^2}{\eta^2}$. Given $\lambda\eta \ne 0$,

$$(a_{ij}^k)^2 \le \max\left(\lambda^2/\eta^2, C^2/\lambda^2, \frac{2\eta C - \lambda^2}{\eta^2}\right) \triangleq C_2,\ 1 \le i,j \le p.$$

Hence,

$$\|A^k\|_F^2 \leq p^2 C_2.$$

The sequence $\{A^k\}$ is uniformly bounded. ■

Lemma 10 *The iteration of Step 1 to Step 3 in Algorithm 1 is a nonexpansive mapping.*

Proof From Lemma 4 and Lemma 6, $\Theta_{\mathcal{B}}$ and Π are nonexpansive mappings. In fact, the composition of nonexpansive mappings is also nonexpansive. So the inner iteration given by Step 2 in Algorithm 1 is nonexpansive. When $k_0 \geq \|X\|_2$, Step 1 in Algorithm 1 is nonexpansive. Again, the composition of Step 1 and Step 2 is nonexpansive. Hence, the iteration of Algorithm 1 is nonexpansive. ■

Lemma 9 and Lemma 10 imply that the mapping of Algorithm 1 is a nonexpansive mapping into a bounded closed convex subset. By Theorem 1 in Browder (1965), it has a fixed point. Then, with all Opial's conditions satisfied, the sequence $\{A^k\}$ has a unique limit point, denoted as A^*, and it is a fixed point of Algorithm 1.

Next, we prove that A^* must be a global minimizer of problem (8). Denote $h(A) = \|A\|_2 - 1$. By Lemma 7 and Lemma 8, A^* satisfies the KKT conditions (Boyd and Vandenberghe, 2004) of problem (16) with $\tau = 1$:

$$\begin{cases} 0 \in A^* - B^0 + \partial P_{\mathcal{B}}(A^*; \lambda/k_0^2, \eta/k_0^2) + \nu^* \partial h(A^*), \\ h(A^*) \leq 0, \\ \nu^* \geq 0, \\ \nu^* h(A^*) = 0. \end{cases}$$

Substituting $B^0 = A^* + \frac{1}{k_0^2}(\Sigma_{XY} - \Sigma_{XX} A^*)$, we have

$$\begin{cases} 0 \in \Sigma_{XY} - \Sigma_{XX} A^* + \partial P_{\mathcal{B}}(A^*; \lambda, \eta) + \tilde{\nu}^* \partial h(A^*), \\ h(A^*) \leq 0, \\ \tilde{\nu}^* \geq 0, \\ \tilde{\nu}^* h(A^*) = 0. \end{cases} \tag{18}$$

Note that problem (8) is convex and its KKT conditions are given by (18). Hence, A^* is a global minimizer of problem (8). The proof is complete.

Appendix B. Proof of Theorem 2

First, we introduce a quantile thresholding rule $\Theta^{\#}(\cdot; m)$ as a variant of the hard thresholding rule. Given $1 \leq m \leq pq$: $A \in \mathbb{R}^{p \times q} \to B \in \mathbb{R}^{p \times q}$ is defined as follows: $b_{ij} = a_{ij}$ if $|a_{ij}|$ is among the m largest in the set of $\{|a_{ij}| : 1 \leq i \leq p, 1 \leq j \leq q\}$, and $b_{ij} = 0$ otherwise.

To prove the function value decreasing property, we introduce the following lemma.

Lemma 11 *$\hat{B} = \Theta^{\#}(A;m)$ is a globally optimal solution to*

$$\min_B l(B) = \frac{1}{2}\|A-B\|_F^2$$
$$s.t.\ \|B\|_0 \le m.$$

Proof Let $I \subset \{(i,j)|1 \le i \le p, 1 \le j \le q\}$ with $|I| = m$. Assuming $B_{I^c} = 0$, we get the optimal solution $\hat{B}$ with $\hat{B} = A_I$. It follows that $l(\hat{B}) = \frac{1}{2}\|A\|_F^2 - \frac{1}{2}\sum_{i,j\in I} a_{ij}^2$. Therefore, the quantile thresholding $\Theta^{\#}(A;m)$ yields a global minimizer. ■

Define a surrogate function

$$\tilde{l}(A,B) = \frac{1}{2}\|Y-XB\|_F^2 + \frac{1}{2}\mathrm{tr}\{(B-A)^{\mathsf{T}}(k_0^2 - X^{\mathsf{T}}X)(B-A)\}.$$

Based on Lemma 11 and $k_0 \ge \|X\|_2$, the function value decreasing property can be proved following the lines of Lemma 8. So we have

$$l(A^k) = \tilde{l}(A^k,A^k) \ge \tilde{l}(A^k,A^{k+1}) = l(A^{k+1}) + \frac{1}{2}(k_0^2 - \|X\|_2^2)\|A^{k+1}-A^k\|_F^2 \ge l(A^{k+1}).$$

The proof is complete.

References

H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

Ya. I. Alber, A. N. Iusem, and M. V. Solodov. On the projected subgradient method for nonsmooth convex optimization in a hilbert space. *Mathematical Programming*, pages 23–35, 1998.

H. H. Bauschke and P. L. Combettes. A dykstra-like algorithm for two monotone operators. *Pacific Journal of Optimization*, 4(3):383–391, Sep. 2008.

T. Blumensath and M. E. Davies. Normalized iterative hard thresholding: Guaranteed stability and performance. *IEEE Journal of Selected Topics in Signal Processing*, 4(2), Apr. 2010.

S. Boyd, N. Parikh, E. Chu, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Information Systems Journal*, 3(1):1–118, 2010.

S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. ISBN 9780521833783.

J. P. Boyle and R. L Dykstra. A method for finding projections onto the intersection of convex sets in hilbert spaces. In *Advances in Order Restricted Statistical Inference*, pages 28–47. Springer, 1986.

F. E. Browder. Nonexpansive nonlinear operators in a banach space. *Proceedings of the National Academy of Sciences of the United States of America*, 54(4):1041, 1965.

F. E. Browder and W. V. Petryshyn. The solution by iteration of nonlinear functional equations in banach spaces. *Bull. Amer. Math. Soc.*, 72:571–575, 1966.

E. Bullmore and O. Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10:186–198, Mar. 2009.

J. V. Burke, A. S. Lewis, and M. L. Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM J. on Optimization*, 15(3):751–779, Mar. 2005. ISSN 1052-6234.

J. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956–1982, Mar. 2010.

F. Curtis and M. Overton. A sequential quadratic programming algorithm for nonconvex, nonsmooth constrained optimization. *SIAM Journal on Optimization*, 22(2):474–500, 2012.

P. Dalgaard. *Introductory Statistics with R, Statistics and Computing*. Springer, Aug. 2008.

S. J. Davis and J. A. Kahn. Interpreting the great moderation: Changes in the volatility of economic activity at the macro and micro levels. *Journal of Economic Perspectives*, 22(4):155–180, 2008.

D.L. Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, Apr. 2006.

R. L. Dykstra. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842, 1983.

B. Efron. Bootstrap methods: another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

J. J. Faith, B. Hayete, J. T. Thaden, I. Mogno, J. Wierzbowski, G. Cottarel, S. Kasif, J. J. Collins, and T. S. Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biol*, 5(1):e8, Jan. 2007.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, Dec. 2001.

J. Fan and R. Li. Statistical challenges with high dimensionality: feature selection in knowledge discovery. In *International Congress of Mathematicans*, Aug. 2006.

J. Fan and J. Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.

J. Fan and J. Lv. A selective overview of variable selection in high dimensional feature space. *Stat Sin.*, 20(1):101–148, Jan. 2010.

J. Fan, R. Samworth, and Y. Wu. Ultrahigh dimensional feature selection: Beyond the linear model. *Journal of Machine Learning Research*, 10:2013–2038, Dec. 2009. ISSN 1532-4435.

C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969.

M. C. Grant and S. P. Boyd. Graph implementations for nonsmooth convex programs. In *Recent Advances in Learning and Control*, pages 95–110. Springer, 2008.

G. E. Halkos and N. G. Tzeremes. Oil consumption and economic efficiency: A comparative analysis of advanced, developing and emerging economies. *Ecological Economics*, 70(7):1354–1362, May 2011.

S. Hanneke, W. Fu, and E. P. Xing. Discrete temporal models of social networks. *Electronic Journal of Statistics*, 4:585–605, 2010.

B.S. He, H. Yang, and S.L. Wang. Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *Journal of Optimization Theory and Applications*, 106 (2):337–356, 2000. ISSN 0022-3239.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

N. Hsu, H. Hung, and Y. Chang. Subset selection for vector autoregressive processes using lasso. *Computational Statistics and Data Analysis*, 52(7):3645–3657, 2008.

P. J. Huber. *Robust Statistics*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 1981.

W. James and C. Stein. Estimation with quadratic loss. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 361–379, 1961.

D. W. Jorgenson and P. J. Wilcoxen. Environmental regulation and U.S. economic growth. *RAND Journal of Economics*, 21(2):314–340, Summer 1990.

H. R. Kunsch. The jackknife and the bootstrap for general stationary observations. *The Annals of Statistics*, 17(3):1217–1241, 1989.

T. G. Lewis. *Network Science: Theory and Applications*. Wiley Publishing, 2009.

Z. Lin. Some software packages for partial SVD computation. *CoRR*, abs/1108.1548, 2011.

H. Lütkepohl. *New Introduction to Multiple Time Series Analysis*. Springer, 2nd edition, 2007.

T.C. Mills and R.N. Markellos. *The Econometric Modelling of Financial Time Series*. Cambridge University Press, 2008.

J. J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Reports of the Paris Academy of Sciences, Series A*, 255:2897–2899, 1962.

M. E. J. Newman and D. J. Watts. *The Structure and Dynamics of Networks*. Princeton University Press, 2006.

Z. Opial. Weak convergence of the sequence of successive approximations for nonexpansive mappings. *Bull. Am. Math. Soc.*, 73:591–597, 1967.

M. L. Overton and R. S. Womersley. On minimizing the spectral radius of a nonsymmetric matrix function - optimality conditions and duality theory. *SIAM J. Matrix Anal. Appl.*, 9(4):473–498, Oct. 1988.

A. B. Owen. A robust hybrid of lasso and ridge regression. *Prediction and Discovery (Contemporary Mathematics)*, 443:59–71, 2007.

D. N. Politis and J. P. Romano. The stationary bootstrap. *Journal of the American Statistical Association*, 89(428):1303–1313, 1994.

G. C. Reinsel. *Elements of Multivariate Time Series Analysis*. New York, Springer, 2nd edition, 1997.

R.T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

Y. She. Thresholding-based iterative selection procedures for model selection and shrinkage. *Electron. J. Statist*, 3:384–415, 2009.

Y. She. An iterative algorithm for fitting nonconvex penalized generalized linear models with grouped predictors. *Computational Statistics and Data Analysis*, 56(10):2976–2990, Oct. 2012.

C. A. Sims. Macroeconomics and reality. *Econometrica*, 48(1):1–48, Jan. 1980.

J. Songsiri and L. Vandenberghe. Topology selection in graphical models of autoregressive processes. *Journal of Machine Learning Research*, 9999:2671–2705, Dec. 2010. ISSN 1532-4435.

J. H. Stock and M. W. Watson. Generalized shrinkage methods for forecasting using many predictors. *Journal of Business & Economic Statistics*, 30(4):481–493, Oct. 2012.

J. F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones, 1998.

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

Y. Tsaig and D. L. Donoho. Extensions of compressed sensing. *Signal Processing*, 86(3):549–571, 2006.

R. H. Tütüncü, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, 95:189–217, 2003.

J. von Neumann. Some matrix inequalities and metrization of matrix space. *Tomsk. Univ. Rev.*, 1: 153–167, 1937.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006.

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 14 (2013) 3105-3128 Submitted 1/12; Revised 4/13; Published 10/13

Algorithms and Hardness Results for Parallel Large Margin Learning

Philip M. Long PLONG@MICROSOFT.COM
Microsoft
1020 Enterprise Way
Sunnyvale, CA 94089

Rocco A. Servedio ROCCO@CS.COLUMBIA.EDU
Department of Computer Science
Columbia University
1214 Amsterdam Ave., Mail Code: 0401
New York, NY 10027

Editor: Yoav Freund

Abstract

We consider the problem of learning an unknown large-margin halfspace in the context of parallel computation, giving both positive and negative results.

As our main positive result, we give a parallel algorithm for learning a large-margin halfspace, based on an algorithm of Nesterov's that performs gradient descent with a momentum term. We show that this algorithm can learn an unknown γ-margin halfspace over n dimensions using $n \cdot \mathrm{poly}(1/\gamma)$ processors and running in time $\tilde{O}(1/\gamma) + O(\log n)$. In contrast, naive parallel algorithms that learn a γ-margin halfspace in time that depends polylogarithmically on n have an inverse quadratic running time dependence on the margin parameter γ.

Our negative result deals with boosting, which is a standard approach to learning large-margin halfspaces. We prove that in the original PAC framework, in which a weak learning algorithm is provided as an oracle that is called by the booster, boosting cannot be parallelized. More precisely, we show that, if the algorithm is allowed to call the weak learner multiple times in parallel within a single boosting stage, this ability does not reduce the overall number of successive stages of boosting needed for learning by even a single stage. Our proof is information-theoretic and does not rely on unproven assumptions.

Keywords: PAC learning, parallel learning algorithms, halfspace learning, linear classifiers

1. Introduction

One of the most fundamental problems in machine learning is learning an unknown halfspace from labeled examples that satisfy a *margin constraint,* meaning that no example may lie too close to the separating hyperplane. In this paper we consider large-margin halfspace learning in the PAC (probably approximately correct) setting of learning from random examples: there is a target halfspace $f(\mathbf{x}) = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x})$, where $\mathbf{w}$ is an unknown unit vector, and an unknown probability distribution $\mathcal{D}$ over the unit ball $\mathbf{B}_n = \{\mathbf{x} \in \mathbf{R}^n : \|\mathbf{x}\|_2 \leq 1\}$ which is guaranteed to have support contained in the set $\{\mathbf{x} \in \mathbf{B}_n : |\mathbf{w} \cdot \mathbf{x}| \geq \gamma\}$ of points that have Euclidean margin at least γ relative to the separating hyperplane. (Throughout this paper we refer to such a combination of target halfspace f and distribution $\mathcal{D}$ as a γ-*margin halfspace*.) The learning algorithm is given access to labeled examples $(\mathbf{x}, f(\mathbf{x}))$

where each $\mathbf{x}$ is independently drawn from $\mathcal{D}$, and it must with high probability output a $(1-\varepsilon)$-accurate hypothesis, that is, a hypothesis $h : \mathbf{R}^n \to \{-1,1\}$ that satisfies $\Pr_{\mathbf{x}\sim\mathcal{D}}[h(\mathbf{x}) \neq f(\mathbf{x})] \leq \varepsilon$.

One of the earliest, and still most important, algorithms in machine learning is the perceptron algorithm (Block, 1962; Novikoff, 1962; Rosenblatt, 1958) for learning a large-margin halfspace. The perceptron is an online algorithm but it can be easily transformed to the PAC setting described above (Vapnik and Chervonenkis, 1974; Littlestone, 1989; Freund and Schapire, 1999); the resulting PAC algorithms run in $\mathrm{poly}(n, \frac{1}{\gamma}, \frac{1}{\varepsilon})$ time, use $O(\frac{1}{\varepsilon\gamma^2})$ labeled examples in $\mathbf{R}^n$, and learn an unknown n-dimensional γ-margin halfspace to accuracy $1-\varepsilon$.

A motivating question: achieving perceptron's performance in parallel? The last few years have witnessed a resurgence of interest in highly efficient parallel algorithms for a wide range of computational problems in many areas including machine learning (Workshop, 2009, 2011). So a natural goal is to develop an efficient parallel algorithm for learning γ-margin halfspaces that matches the performance of the perceptron algorithm. A well-established theoretical notion of efficient parallel computation (see, for example, the text by Greenlaw et al. (1995) and the many references therein) is that an efficient parallel algorithm for a problem with input size N is one that uses $\mathrm{poly}(N)$ processors and runs in parallel time $\mathrm{polylog}(N)$. Since the input to the perceptron algorithm is a sample of $\mathrm{poly}(\frac{1}{\varepsilon}, \frac{1}{\gamma})$ labeled examples in $\mathbf{R}^n$, we naturally arrive at the following:

> **Main Question:** Is there a learning algorithm that uses $\mathrm{poly}(n, \frac{1}{\gamma}, \frac{1}{\varepsilon})$ processors and runs in time $\mathrm{poly}(\log n, \log\frac{1}{\gamma}, \log\frac{1}{\varepsilon})$ to learn an unknown n-dimensional γ-margin halfspace to accuracy $1-\varepsilon$?

Following Vitter and Lin (1992), we use a CRCW PRAM model of parallel computation. This abstracts away issues like communication and synchronization, allowing us to focus on the most fundamental issues. Also, as did Vitter and Lin (1992), we require that an efficient parallel learning algorithm's hypothesis must be efficiently evaluatable in parallel, since otherwise all the computation required to run any polynomial-time learning algorithm could be "offloaded" onto evaluating the hypothesis. Because halfspace learning algorithms may be sensitive to issues of numerical precision, these are not abstracted away in our model; we assume that numbers are represented as rationals.

As noted by Freund (1995) (see also Lemma 2 below), the existence of efficient boosting algorithms such as the algorithms of Freund (1995) and Schapire (1990) implies that any PAC learning algorithm can be efficiently parallelized in terms of its dependence on the accuracy parameter ε: more precisely, any PAC learnable class C of functions can be PAC learned to accuracy $1-\varepsilon$ using $O(1/\varepsilon)$ processors by an algorithm whose running time dependence on ε is $O(\log\left(\frac{1}{\varepsilon}\right) \cdot \mathrm{poly}(\log\log(1/\varepsilon)))$, by boosting an algorithm that learns to accuracy (say) $9/10$. We may thus equivalently restate the above question as follows.

> **Main Question (simplified):** Is there a learning algorithm that uses $\mathrm{poly}(n, \frac{1}{\gamma})$ processors and runs in time $\mathrm{poly}(\log n, \log\frac{1}{\gamma})$ to learn an unknown n-dimensional γ-margin halfspace to accuracy $9/10$?

The research reported in this paper is inspired by this question, which we view as a fundamental open problem about the abilities and limitations of efficient parallel learning algorithms.

Algorithm	No. processors	Running time
naive parallelization of perceptron	$\mathrm{poly}(n, 1/\gamma)$	$\tilde{O}(1/\gamma^2) + O(\log n)$
(Servedio, 2003)	$\mathrm{poly}(n, 1/\gamma)$	$\tilde{O}(1/\gamma^2) + O(\log n)$
poly-time linear programming (Blumer et al., 1989)	1	$\mathrm{poly}(n, \log(1/\gamma))$
this paper (algorithm of Section 2)	$n \cdot \mathrm{poly}(1/\gamma)$	$\tilde{O}(1/\gamma) + O(\log n)$

Table 1: Bounds on various parallel algorithms for learning a γ-margin halfspace over $\mathbf{R}^n$.

1.1 Relevant Prior Results

Table 1 summarizes the running time and number of processors used by various parallel algorithms to learn a γ-margin halfspace over $\mathbf{R}^n$.

The naive parallelization of perceptron in the first line of the table is an algorithm that runs for $O(1/\gamma^2)$ stages. In each stage it processes all of the $O(1/\gamma^2)$ examples simultaneously in parallel, identifies one that causes the perceptron algorithm to update its hypothesis vector, and performs this update. Since the examples are n-dimensional this can be accomplished in $O(\log(n/\gamma))$ time using $O(n/\gamma^2)$ processors; the mistake bound of the online perceptron algorithm is $1/\gamma^2$, so this gives a running time bound of $\tilde{O}(1/\gamma^2) \cdot \log n$. We do not see how to obtain parallel time bounds better than $O(1/\gamma^2)$ from recent analyses of other algorithms based on gradient descent (Collins et al., 2002; Dekel et al., 2011; Bradley et al., 2011), some of which use assumptions incomparable in strength to the γ-margin condition studied here.

The second line of the table corresponds to a similar naive parallelization of the boosting-based algorithm of Servedio (2003) that achieves perceptron-like performance for learning a γ-margin halfspace. This algorithm boosts for $O(1/\gamma^2)$ stages over a $O(1/\gamma^2)$-size sample. At each stage of boosting this algorithm computes a real-valued weak hypothesis based on the vector average of the (normalized) examples weighted according to the current distribution; since the sample size is $O(1/\gamma^2)$ this can be done in $O(\log(n/\gamma))$ time using $\mathrm{poly}(n, 1/\gamma)$ processors. Since the boosting algorithm runs for $O(1/\gamma^2)$ stages, the overall running time bound is $\tilde{O}(1/\gamma^2) \cdot \log n$. (For both this algorithm and the perceptron the time bound can be improved to $\tilde{O}(1/\gamma^2) + O(\log n)$ as claimed in the table by using an initial random projection step. We show how to do this in Section 2.3.)

The third line of the table, included for comparison, is simply a standard sequential algorithm for learning a halfspace based on polynomial-time linear programming executed on one processor (Blumer et al., 1989; Karmarkar, 1984).

In addition to the results summarized in the table, we note that efficient parallel algorithms have been developed for some simpler PAC learning problems such as learning conjunctions, disjunctions, and symmetric Boolean functions (Vitter and Lin, 1992). Bshouty et al. (1998) gave efficient parallel PAC learning algorithms for some geometric constant-dimensional concept classes. Collins et al. (2002) presented a family of boosting-type algorithms that optimize Bregman divergences by updating a collection of parameters in parallel; however, their analysis does not seem to imply that the algorithms need fewer than $\Omega(1/\gamma^2)$ stages to learn γ-margin halfspaces.

In terms of negative results for parallel learning, Vitter and Lin (1992) showed that (under a complexity-theoretic assumption) there is no parallel algorithm using $\mathrm{poly}(n)$ processors and $\mathrm{polylog}(n)$ time that constructs a halfspace hypothesis that is consistent with a given linearly separable data set of n-dimensional labeled examples. This does not give a negative answer to the

main question for several reasons: first, the main question allows any hypothesis representation that can be efficiently evaluated in parallel, whereas the hardness result requires the hypothesis to be a halfspace. Second, the main question allows the algorithm to use $\text{poly}(n, 1/\gamma)$ processors and to run in $\text{poly}(\log n, \log \frac{1}{\gamma})$ time, whereas the hardness result of Vitter and Lin (1992) only rules out algorithms that use $\text{poly}(n, \log \frac{1}{\gamma})$ processors and run in $\text{poly}(\log n, \log\log \frac{1}{\gamma})$ time. Finally, the main question allows the final hypothesis to err on up to (say) 5% of the points in the data set, whereas the hardness result of Vitter and Lin (1992) applies only to algorithms whose hypotheses correctly classify all points in the data set.

Finally, we note that the main question has an affirmative answer if it is restricted so that either the number of dimensions n or the margin parameter γ is fixed to be a constant (so the resulting restricted question asks whether there is an algorithm that uses polynomially many processors and polylogarithmic time in the remaining parameter). If γ is fixed to a constant then either of the first two entries in Table 1 gives a $\text{poly}(n)$-processor, $O(\log n)$-time algorithm. If n is fixed to a constant then the efficient parallel algorithm of Alon and Megiddo (1994) for linear programming in constant dimension can be used to learn a γ-margin halfspace using $\text{poly}(1/\gamma)$ processors in $\text{polylog}(1/\gamma)$ running time (see also Vitter and Lin, 1992, Theorem 3.4).

1.2 Our Results

We give positive and negative results on learning halfspaces in parallel that are inspired by the main question stated above.

1.2.1 POSITIVE RESULTS

Our main positive result is a parallel algorithm for learning large-margin halfspaces, based on a rapidly converging gradient method due to Nesterov (2004), which uses $O(n \cdot \text{poly}(1/\gamma))$ processors to learn γ-margin halfspaces in parallel time $\tilde{O}(1/\gamma) + O(\log n)$ (see Table 1). (An earlier version of this paper (Long and Servedio, 2011) analyzed on algorithm based on interior-point methods from convex optimization and fast parallel algorithms for linear algebra, showing that it uses $\text{poly}(n, 1/\gamma)$ processors to learn γ-margin halfspaces in parallel time $\tilde{O}(1/\gamma) + O(\log n)$.) We are not aware of prior parallel algorithms that provably learn γ-margin halfspaces running in time polylogarithmic in n and subquadratic in $1/\gamma$.

We note that simultaneously and independently of the initial conference publication of our work (Long and Servedio, 2011), Soheili and Peña (2012) proposed a variant of the perceptron algorithm and shown that it terminates in $O\left(\frac{\sqrt{\log n}}{\gamma}\right)$ iterations rather than the $1/\gamma^2$ iterations of the original perceptron algorithm. Like our algorithm, the Soheili and Peña (2012) algorithm uses ideas of Nesterov (2005). Soheili and Peña (2012) do not discuss a parallel implementation of their algorithm, but since their algorithm performs an n-dimensional matrix-vector multiplication at each iteration, it appears that a parallel implementation of their algorithm would use $\Omega(n^2)$ processors and would have parallel running time at least $\Omega\left(\frac{(\log n)^{3/2}}{\gamma}\right)$ (assuming that multiplying a $n \times n$ matrix by an $n \times 1$ vector takes parallel time $\Theta(\log n)$ using n^2 processors). In contrast, our algorithm requires a linear number of processors as a function of n, and has parallel running time $\tilde{O}(1/\gamma) + O(\log n)$.[1]

1. We note also that Soheili and Peña (2012) analyze the number of iterations of their algorithm, and not the computation time. In particular, they do not deal with finite precision issues, whereas a significant portion of our analysis concerns

1.2.2 NEGATIVE RESULTS

By modifying our analysis of the algorithm we present, we believe that it may be possible to establish similar positive results for other formulations of the large-margin learning problem, including ones (see Shalev-Shwartz and Singer, 2010) that have been tied closely to weak learnability. In contrast, our main negative result is an information-theoretic argument that suggests that such positive parallel learning results cannot be obtained by boosting alone. We show that in a framework where the weak learning algorithm must be invoked as an oracle, boosting cannot be parallelized: being able to call the weak learner multiple times in parallel within a single boosting stage does not reduce the overall number of sequential stages of boosting that are required. We prove that any parallel booster must perform $\Omega(\log(1/\varepsilon)/\gamma^2)$ sequential stages of boosting a "black-box" γ-advantage weak learner to learn to accuracy $1-\varepsilon$ in the worst case; this extends an earlier $\Omega(\log(1/\varepsilon)/\gamma^2)$ lower bound of Freund (1995) for standard (sequential) boosters that can only call the weak learner once per stage.

2. An Algorithm Based on Nesterov's Algorithm

In this section we describe and analyze a parallel algorithm for learning a γ-margin halfspace. The algorithm of this section applies an algorithm of Nesterov (2004) that, roughly speaking, approximately minimizes a suitably smooth convex function to accuracy ε using $O(\sqrt{1/\varepsilon})$ iterative steps (Nesterov, 2004), each of which can be easily parallelized.

Directly applying the basic Nesterov algorithm gives us an algorithm that uses $O(n)$ processors, runs in parallel time $O(\log(n)\cdot(1/\gamma))$, and outputs a halfspace hypothesis that has constant accuracy. By combining the basic algorithm with random projection and boosting we get the following stronger result:

Theorem 1 *There is a parallel algorithm with the following performance guarantee: Let $f, \mathcal{D}$ define an unknown γ-margin halfspace over $\mathbf{B}_n$. The algorithm is given as input $\varepsilon, \delta > 0$ and access to labeled examples $(\mathbf{x}, f(\mathbf{x}))$ that are drawn independently from $\mathcal{D}$. It runs in*

$$O(((1/\gamma)\text{polylog}(1/\gamma)+\log(n))\log(1/\varepsilon)\text{poly}(\log\log(1/\varepsilon))+\log\log(1/\delta))$$

parallel time, uses

$$n\cdot\text{poly}(1/\gamma, 1/\varepsilon, \log(1/\delta))$$

processors, and with probability $1-\delta$ it outputs a hypothesis h satisfying $\Pr_{\mathbf{x}\sim\mathcal{D}}[h(\mathbf{x})\neq f(\mathbf{x})]\leq\varepsilon$.

We assume that the value of γ is "known" to the algorithm, since otherwise the algorithm can use a standard "guess and check" approach trying $\gamma = 1, 1/2, 1/4$, etc., until it finds a value that works.

Freund (1995) indicated how to parallelize his boosting-by-filtering algorithm. In Appendix A, we provide a detailed proof of the following lemma.

Lemma 2 (Freund, 1995) *Let $\mathcal{D}$ be a distribution over (unlabeled) examples. Let A be a parallel learning algorithm, and c_δ and c_ε be absolute positive constants, such that for all $\mathcal{D}'$ with*

such issues, in order to fully establish our claimed bounds on the number of processors and the parallel running time of our algorithms.

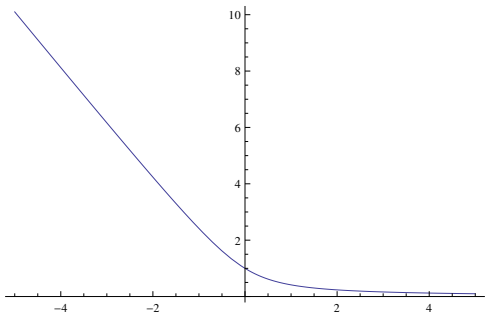


Figure 1: A plot of a loss function ϕ used in Section 2.

support$(\mathcal{D}') \subseteq$ *support*$(\mathcal{D})$*, given draws* $(x, f(x))$ *from* $\mathcal{D}'$*, with probability* c_δ*, A outputs a hypothesis with accuracy* $\frac{1}{2} + c_\varepsilon$ *(w.r.t.* $\mathcal{D}'$*) using* $\mathcal{P}$ *processors in* $\mathcal{T}$ *time. Then there is a parallel algorithm B that, given access to independent labeled examples* $(\boldsymbol{x}, f(\boldsymbol{x}))$ *drawn from* $\mathcal{D}$*, with probability* $1-\delta$*, constructs a* $(1-\varepsilon)$*-accurate hypothesis (w.r.t.* $\mathcal{D}$*) in* $O(\mathcal{T}\log(1/\varepsilon)\mathrm{poly}(\log\log(1/\varepsilon)) + \log\log(1/\delta))$ *time using* $\mathrm{poly}(\mathcal{P}, 1/\varepsilon, \log(1/\delta))$ *processors.*

In Section 2.1 we describe the basic way that Nesterov's algorithm can be used to find a halfspace hypothesis that approximately minimizes a smooth loss function over a set of γ-margin labeled examples. (This section has nothing to do with parallelism.) Then later we explain how this algorithm is used in the larger context of a parallel algorithm for halfspaces.

2.1 The Basic Algorithm

Let $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ be a data set of m examples labeled according to the target γ-margin halfspace f; that is, $y_i = f(\mathbf{x}_i)$ for all i.

We will apply Nesterov's algorithm to minimize a regularized loss as follows.
The loss part. For $z \in \mathbf{R}$ we define

$$\phi(z) = \sqrt{1+z^2} - z.$$

(See Figure 1 for a plot of ϕ.) For $\mathbf{v} \in \mathbf{R}^n$ we define

$$\Phi(\mathbf{v}) = \frac{1}{m}\sum_{t=1}^{m} \phi(y_t(\mathbf{v}\cdot\mathbf{x}_t)).$$

The regularization part. We define a regularizer

$$R(\mathbf{v}) = \gamma^2\|\mathbf{v}\|^2/100$$

where $\|\cdot\|$ denotes the 2-norm.

We will apply Nesterov's iterative algorithm to minimize the following function

$$\Psi(\mathbf{v}) = \Phi(\mathbf{v}) + R(\mathbf{v}).$$

Let $g(\mathbf{v})$ be the gradient of Ψ at $\mathbf{v}$. We will use the following algorithm, due to Nesterov (2004) (see Section 2.2.1), which we call A_{Nes}. The algorithm takes a single input parameter $\gamma > 0$.

Algorithm A_{Nes}:

- Set $\mu = \gamma^2/50$, $L = 51/50$.
- Initialize $\mathbf{v}_0 = \mathbf{z}_0 = \mathbf{0}$.
- For each $k = 0, 1, \ldots$, set
  - $\mathbf{v}_{k+1} = \mathbf{z}_k - \frac{1}{L} g(\mathbf{z}_k)$, and
  - $\mathbf{z}_{k+1} = \mathbf{v}_{k+1} + \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(\mathbf{v}_{k+1} - \mathbf{v}_k)$.

We begin by establishing various bounds on Ψ that Nesterov uses in his analysis of A_{Nes}.

Lemma 3 *The gradient g of Ψ has a Lipschitz constant at most $51/50$.*

Proof: We have

$$\frac{\partial \Psi}{\partial v_i} = \frac{1}{m}\sum_t \phi'(y_t(\mathbf{v}\cdot\mathbf{x}_t))y_t x_{t,i} + \gamma^2 v_i/50$$

and hence, writing $g(\mathbf{v})$ to denote the gradient of Ψ at $\mathbf{v}$, we have

$$g(\mathbf{v}) = \frac{1}{m}\sum_t \phi'(y_t(\mathbf{v}\cdot\mathbf{x}_t))y_t\mathbf{x}_t + \gamma^2\mathbf{v}/50.$$

Choose $\mathbf{r} \in \mathbf{R}^n$. Applying the triangle inequality, we have

$$\begin{aligned}
||g(\mathbf{v}) - g(\mathbf{r})|| &= \left\| \frac{1}{m}\sum_t (\phi'(y_t(\mathbf{v}\cdot\mathbf{x}_t)) - \phi'(y_t(\mathbf{r}\cdot\mathbf{x}_t)))y_t\mathbf{x}_t + \gamma^2(\mathbf{v}-\mathbf{r})/50 \right\| \\
&\leq \frac{1}{m}\sum_t ||(\phi'(y_t(\mathbf{v}\cdot\mathbf{x}_t)) - \phi'(y_t(\mathbf{r}\cdot\mathbf{x}_t)))y_t\mathbf{x}_t|| + \gamma^2||\mathbf{v}-\mathbf{r}||/50 \\
&\leq \frac{1}{m}\sum_t |\phi'(y_t(\mathbf{v}\cdot\mathbf{x}_t)) - \phi'(y_t(\mathbf{r}\cdot\mathbf{x}_t))| + \gamma^2||\mathbf{v}-\mathbf{r}||/50,
\end{aligned}$$

since each vector $\mathbf{x}_t$ has length at most 1. Basic calculus gives that ϕ'' is always at most 1, and hence

$$|\phi'(y_t(\mathbf{v}\cdot\mathbf{x}_t)) - \phi'(y_t(\mathbf{r}\cdot\mathbf{x}_t))| \leq |\mathbf{v}\cdot\mathbf{x}_t - \mathbf{r}\cdot\mathbf{x}_t| \leq ||\mathbf{v}-\mathbf{r}||,$$

again since $\mathbf{x}_t$ has length at most 1. The bound then follows from the fact that $\gamma^2 \leq 1$. ■

We recall the definition of strong convexity (Nesterov, 2004, pp. 63–64): a multivariate function q is μ-strongly convex if for all $\mathbf{v}, \mathbf{w}$ and all $\alpha \in [0,1]$ it holds that

$$q(\alpha\mathbf{v} + (1-\alpha)\mathbf{w}) \leq \alpha q(\mathbf{v}) + (1-\alpha)q(\mathbf{w}) - \frac{\mu\alpha(1-\alpha)||\mathbf{v}-\mathbf{w}||^2}{2}.$$

(For intuition's sake, it may be helpful to note that a suitably smooth q is μ-strongly convex if any restriction of q to a line has second derivative that is always at least μ.) We recall the fact that strongly convex functions have unique minimizers.

Lemma 4 *Ψ is μ-strongly convex.*

Proof: This follows directly from the fact that $\mu = \gamma^2/50$, Φ is convex, and $||\mathbf{v}||^2$ is 2-strongly convex. ■

Given the above, the following lemma is an immediate consequence of Theorem 2.2.3 of Nesterov's (2004) book. The lemma upper bounds the difference between $\Psi(\mathbf{v}_k)$, where $\mathbf{v}_k$ is the point computed in the k-th iteration of Nesterov's algorithm A_{Nes}, and the true minimum value of Ψ. A proof is in Appendix B.

Lemma 5 *Let* $\mathbf{w}$ *be the minimizer of* Ψ. *For each k, we have* $\Psi(\mathbf{v}_k) - \Psi(\mathbf{w}) \leq \frac{4L(1+\mu||\mathbf{w}||^2/2)}{(2\sqrt{L}+k\sqrt{\mu})^2}$.

2.2 The Finite Precision Algorithm

The algorithm analyzed in the previous subsection computes real numbers with infinite precision. Now we will analyze a finite precision variant of the algorithm, which we call A_{Nfp} (for "Nesterov finite precision").

(We note that d'Aspremont 2008, also analyzed a similar algorithm with an approximate gradient, but we were not able to apply his results in our setting because of differences between his assumptions and our needs. For example, the algorithm described by d'Aspremont 2008, assumed that optimization was performed over a compact set C, and periodically projected solutions onto C; it was not obvious to us how to parallelize this algorithm.)

We begin by writing the algorithm as if it took two parameters, γ and a precision parameter $\beta > 0$. The analysis will show how to set β as a function of γ. To distinguish between A_{Nfp} and A_{Nes} we use hats throughout our notation below.

Algorithm A_{Nfp}**:**

- Set $\mu = \gamma^2/50$, $L = 51/50$.
- Initialize $\hat{\mathbf{v}}_0 = \hat{\mathbf{z}}_0 = \mathbf{0}$.
- For each $k = 0, 1, \ldots,$
  - Let $\hat{\mathbf{r}}_k$ be such that $||\hat{\mathbf{r}}_k - \frac{1}{L}g(\hat{\mathbf{z}}_k)|| \leq \beta$. Set
  - $\hat{\mathbf{v}}_{k+1} = \hat{\mathbf{z}}_k - \hat{\mathbf{r}}_k$, and
  - $\hat{\mathbf{z}}_{k+1} = \hat{\mathbf{v}}_{k+1} + \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(\hat{\mathbf{v}}_{k+1} - \hat{\mathbf{v}}_k)$.

We discuss the details of exactly how this finite-precision algorithm is implemented, and the parallel running time required for such an implementation, at the end of this section.

Our analysis of this algorithm will proceed by quantifying how closely its behavior tracks that of the infinite-precision algorithm.

Lemma 6 *Let* $\mathbf{v}_0, \mathbf{v}_1, \ldots$ *be the sequence of points computed by the infinite precision version of Nesterov's algorithm, and* $\hat{\mathbf{v}}_0, \hat{\mathbf{v}}_1, \ldots$ *be the corresponding finite-precision sequence. Then for all k, we have* $||\mathbf{v}_k - \hat{\mathbf{v}}_k|| \leq \beta \cdot 7^k$.

Proof: Let $\hat{\mathbf{s}}_k = \hat{\mathbf{r}}_k - g(\hat{\mathbf{z}}_k)$. Our proof is by induction, with the additional inductive hypothesis that $||\mathbf{z}_k - \hat{\mathbf{z}}_k|| \leq 3\beta \cdot 7^k$.

The base case is trivially true.

We have

$$||\mathbf{v}_{k+1} - \hat{\mathbf{v}}_{k+1}|| = \left|\left|(\mathbf{z}_k - \frac{1}{L}g(\mathbf{z}_k)) - (\hat{\mathbf{z}}_k - (\frac{1}{L}g(\hat{\mathbf{z}}_k) + \hat{\mathbf{s}}_k)\right|\right|,$$

and, using the triangle inequality, we get

$$\begin{aligned}
||\mathbf{v}_{k+1} - \hat{\mathbf{v}}_{k+1}|| &\leq ||\mathbf{z}_k - \hat{\mathbf{z}}_k|| + \left|\left|\frac{1}{L}g(\mathbf{z}_k) - (\frac{1}{L}g(\hat{\mathbf{z}}_k) + \hat{\mathbf{s}}_k)\right|\right| \\
&\leq 3\beta \cdot 7^k + \left|\left|\frac{1}{L}g(\mathbf{z}_k) - (\frac{1}{L}g(\hat{\mathbf{z}}_k) + \hat{\mathbf{s}}_k)\right|\right| \\
&\leq 3\beta \cdot 7^k + \frac{1}{L}||g(\mathbf{z}_k) - g(\hat{\mathbf{z}}_k)|| + ||\hat{\mathbf{s}}_k|| \qquad \text{(triangle inequality)} \\
&\leq 3\beta \cdot 7^k + ||\mathbf{z}_k - \hat{\mathbf{z}}_k|| + ||\hat{\mathbf{s}}_k|| \qquad \text{(by Lemma 3)} \\
&\leq 3\beta \cdot 7^k + 3\beta \cdot 7^k + \beta \qquad \text{(by definition of } \hat{\mathbf{s}}_k) \\
&< \beta \cdot 7^{k+1}.
\end{aligned}$$

Also, we have

$$\begin{aligned}
||\mathbf{z}_{k+1} - \hat{\mathbf{z}}_{k+1}|| &= \left|\left|\frac{2}{1+\sqrt{\mu/L}}(\mathbf{v}_{k+1} - \hat{\mathbf{v}}_{k+1}) - \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(\mathbf{v}_k - \hat{\mathbf{v}}_k)\right|\right| \\
&\leq \left|\left|\frac{2}{1+\sqrt{\mu/L}}(\mathbf{v}_{k+1} - \hat{\mathbf{v}}_{k+1})\right|\right| + \left|\left|\frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(\mathbf{v}_k - \hat{\mathbf{v}}_k)\right|\right| \\
&\leq 2||\mathbf{v}_{k+1} - \hat{\mathbf{v}}_{k+1}|| + ||\mathbf{v}_k - \hat{\mathbf{v}}_k|| \\
&\leq 2\beta \cdot 7^{k+1} + \beta \cdot 7^k \\
&\leq 3\beta \cdot 7^{k+1},
\end{aligned}$$

completing the proof. ■

2.3 Application to Learning

Now we are ready to prove Theorem 1. By Lemma 2 it suffices to prove the theorem in the case in which $\varepsilon = 7/16$ and $\delta = 1/2$.

We may also potentially reduce the number of variables by applying a random projection. We say that a *random projection matrix* is a matrix A chosen uniformly from $\{-1,1\}^{n\times d}$. Given such an A and a unit vector $\mathbf{w} \in \mathbf{R}^n$ (defining a target halfspace $f(\mathbf{x}) = \text{sign}(\mathbf{w}\cdot\mathbf{x})$), let $\mathbf{w}'$ denote the vector $(1/\sqrt{d})\mathbf{w}A \in \mathbf{R}^d$. After transformation by A the distribution $\mathcal{D}$ over $\mathbf{B}_n$ is transformed to a distribution $\mathcal{D}'$ over $\mathbf{R}^d$ in the natural way: a draw $\mathbf{x}'$ from $\mathcal{D}'$ is obtained by making a draw $\mathbf{x}$ from $\mathcal{D}$ and setting $\mathbf{x}' = (1/\sqrt{d})\mathbf{x}A$. We will use the following lemma, which is a slight variant of known lemmas (Arriaga and Vempala, 2006; Blum, 2006); we prove this exact statement in Appendix C.

Lemma 7 *Let $f(\mathbf{x}) = \text{sign}(\mathbf{w}\cdot\mathbf{x})$ and $\mathcal{D}$ define a γ-margin halfspace as described in the introduction. For $d = O((1/\gamma^2)\log(1/\gamma))$, a random $n \times d$ projection matrix A will with probability $99/100$ induce $\mathcal{D}'$ and $\mathbf{w}'$ as described above such that $\Pr_{\mathbf{x}'\sim\mathcal{D}'}\left[\left|\frac{\mathbf{w}'}{||\mathbf{w}'||}\cdot\mathbf{x}'\right| < \gamma/2 \quad or \quad ||\mathbf{x}'||_2 > 2\right] \leq \gamma^4$.*

We assume without loss of generality that $\gamma = 1/\text{integer}$.

The algorithm first selects an $n \times d$ random projection matrix A where $d = O(\log(1/\gamma)/\gamma^2)$. This defines a transformation $\Phi_A : \mathbf{B}_n \to \mathbf{R}^d$ as follows: given $\mathbf{x} \in \mathbf{B}_n$, the vector $\Phi_A(\mathbf{x}) \in \mathbf{R}^d$ is obtained by

(i) rounding each $\mathbf{x}_i$ to the nearest integer multiple of $1/(4\lceil\sqrt{n/\gamma}\rceil)$; then

(ii) setting $\mathbf{x}' = \left(\frac{1}{2\sqrt{d}}\right)\mathbf{x}A$ (we scale down by an additional factor of two to get examples that are contained in the unit ball $\mathbf{B}_d$); and finally

(iii) rounding each $\mathbf{x}'_i$ to the nearest multiple of $1/(8\lceil d/\gamma\rceil)$.

Given $\mathbf{x}$ it is easy to compute $\Phi_A(\mathbf{x})$ using $O(n\log(1/\gamma)/\gamma^2)$ processors in $O(\log(n/\gamma))$ time. Let $\mathcal{D}'$ denote the distribution over $\mathbf{R}^d$ obtained by applying Φ_A to $\mathcal{D}$. Across all coordinates $\mathcal{D}'$ is supported on rational numbers with the same poly$(1/\gamma)$ common denominator. By Lemma 7, with probability $99/100$

$$\Pr_{\mathbf{x}' \sim \mathcal{D}'}\left[\left|\mathbf{x}' \cdot (\mathbf{w}'/\|\mathbf{w}'\|)\right| < \gamma' \stackrel{\text{def}}{=} \gamma/8 \quad \text{or} \quad \|\mathbf{x}'\|_2 > 1\right] \leq \gamma^4.$$

Our algorithm draws $c_0 d$ examples by sampling from $\mathcal{D}'$. Applying Lemma 7, we may assume without loss of generality that our examples have $d = O(\log(1/\gamma)/\gamma^2)$ and that the margin γ' after the projection is at least $\Theta(\gamma)$, and that all the coordinates of all the examples have a common denominator which is at most poly$(1/\gamma)$. Thus far the algorithm has used $O(\log(n/\gamma))$ parallel time and $O(n\log(1/\gamma)/\gamma^2)$ many processors.

Next, the algorithm applies A_{Nfp} from the previous section for K stages, where $K = \lceil c_1/\gamma'\rceil$ and $\beta = c_2 7^{-K}$. Here c_0, c_1, and c_2 are absolute positive constants; our analysis will show that there exist choices of these constants that give Theorem 1.

For our analysis, as before, let $\mathbf{w}$ be the minimizer of Ψ, and let $\mathbf{u}$ be a unit normal vector for the target halfspace $f(\mathbf{x}) = \text{sign}(\mathbf{u} \cdot \mathbf{x})$. (We emphasize that Ψ is now defined using the projected d-dimensional examples and with γ' in place of γ in the definition of the regularizer R.)

Our first lemma gives an upper bound on the optimal value of the objective function.

Lemma 8 $\Psi(\mathbf{w}) \leq 0.26$.

Proof Since $\mathbf{w}$ is the minimizer of Ψ we have $\Psi(\mathbf{w}) \leq \Psi(3\mathbf{u}/\gamma')$. In turn $\Psi(3\mathbf{u}/\gamma')$ is easily seen to be at most $\phi(3) + 9/100 \leq 0.26$, since every example has margin at least γ' with respect to $\mathbf{u}$. ■

Next, we bound the norm of $\mathbf{w}$.

Lemma 9 $||\mathbf{w}||^2 \leq 26/\gamma'^2$.

Proof: The definition of Ψ gives

$$||\mathbf{w}||^2 \leq 100\Psi(\mathbf{w})/\gamma'^2$$

and combining with Lemma 8 gives $||\mathbf{w}||^2 \leq 26/\gamma'^2$. ■

Now we can bound the objective function value of $\mathbf{v}_K$.

Lemma 10 *For c_1 a sufficiently large absolute constant, we have $\Psi(\mathbf{v}_K) \leq 2/5$.*

Proof: Plugging Lemma 9 into the RHS of Lemma 5 and simplifying, we get

$$\Psi(\mathbf{v}_K) - \Psi(\mathbf{w}) \leq \frac{751}{25(2\sqrt{51}+\gamma' K)^2}.$$

Applying Lemma 8, we get

$$\Psi(\mathbf{v}_K) \leq 0.26 + \frac{751}{25(2\sqrt{51}+\gamma' K)^2}.$$

from which the lemma follows. ■

Now we can bound $\mathbf{v}_K$ nearly the same way that we bounded $\mathbf{w}$:

Lemma 11 $||\mathbf{v}_K|| \leq 7/\gamma'$.

Proof: The argument is similar to the proof of Lemma 9, using Lemma 10 in place of Lemma 8. ■

Now we can bound the value of the objective function of the finite precision algorithm.

Lemma 12 *There exist absolute positive constants c_1, c_2 such that $\Psi(\hat{\mathbf{v}}_K) \leq 3/7$.*

Proof Because $\beta = c_2 7^{-\lceil c_1/\gamma' \rceil}$, Lemma 6 implies that $||\hat{\mathbf{v}}_K - \mathbf{v}_K|| \leq c_2$. Since ϕ has a Lipschitz constant of 2, so does Φ, and consequently we have that

$$\Phi(\hat{\mathbf{v}}_K) - \Phi(\mathbf{v}_K) \leq 2c_2. \tag{1}$$

Next, since Lemma 11 gives $||\mathbf{v}_K|| \leq 7/\gamma'$, and $||\hat{\mathbf{v}}_K - \mathbf{v}_K|| \leq c_2$, we have $||\hat{\mathbf{v}}_K|| \leq 7/\gamma' + c_2$, which in turn implies

$$\|\hat{\mathbf{v}}_K\|^2 - \|\mathbf{v}_K\|^2 \leq (7/\gamma' + c_2)^2 - (7/\gamma')^2 = 14c_2/\gamma' + c_2^2.$$

and thus

$$R(\hat{\mathbf{v}}_K) - R(\mathbf{v}_K) \leq \frac{14c_2\gamma'}{100} + \frac{(\gamma')^2 c_2^2}{100}.$$

Combining this with (1), we get that for c_2 less than a sufficiently small positive absolute constant, we have $\Psi(\hat{\mathbf{v}}_K) - \Psi(\mathbf{v}_K) < 3/7 - 2/5$, and combining with Lemma 10 completes the proof. ■

Finally, we observe that $\Psi(\hat{\mathbf{v}}_k)$ is an upper bound on the fraction of examples in the sample that are misclassified by $\hat{\mathbf{v}}_k$. Taking c_0 sufficiently large and applying standard VC sample complexity bounds, we have established the (ε, δ) PAC learning properties of the algorithm. (Recall from the start of this subsection that we have taken $\varepsilon = 7/16$ and $\delta = 1/2$.)

It remains to analyze the parallel time complexity of the algorithm. We have already analyzed the parallel time complexity of the initial random projection stage, and shown that we may take the finite-precision iterative algorithm A_{Nfp} to run for $O(1/\gamma)$ stages, so it suffices to analyze the parallel time complexity of each stage A_{Nfp}. We will show that each stage runs in parallel time $\text{polylog}(1/\gamma)$ and thus establish the theorem.

Recall that we have set $\beta = \Theta(7^{-K})$ and that $K = \Theta(1/\gamma)$. The invariant we maintain throughout each iteration k of algorithm A_{Nfp} is that each coordinate of $\hat{\mathbf{v}}_k$ is a $\text{poly}(K)$-bit rational number and each coordinate of $\hat{\mathbf{z}}_k$ is a $\text{poly}(K)$-bit rational number. It remains to show that given such values $\hat{\mathbf{v}}_k$ and $\hat{\mathbf{z}}_k$, in parallel time $\text{polylog}(1/\gamma)$ using $\log(1/\gamma)$ processors,

1. it is possible to compute each coordinate $g(\hat{\mathbf{z}}_k)_i$ to accuracy $2^{-100K}/\sqrt{d}$;
2. it is possible to determine a vector $\hat{\mathbf{r}}_k$ such that $\|\hat{\mathbf{r}}_k - g(\hat{\mathbf{z}}_k)\| \leq \beta$, and that each coefficient of the new value $\hat{\mathbf{v}}_{k+1} = \hat{\mathbf{z}}_k - \hat{\mathbf{r}}_k$ is again a poly(K)-bit rational number; and
3. it is possible to compute the new value $\hat{\mathbf{z}}_{k+1}$ and that each coordinate of $\hat{\mathbf{z}}_{k+1}$ is again a poly(K)-bit rational number.

We begin by analyzing the approximate computation of the gradient. Recall that

$$g(\mathbf{v}) = \frac{1}{m}\sum_t \phi'(y_t(\mathbf{v}\cdot\mathbf{x}_t))y_t\mathbf{x}_t + \gamma^2\mathbf{v}/50.$$

Note that

$$\phi'(z) = \frac{z}{\sqrt{1+z^2}} - 1.$$

To analyze the approximation of ϕ' we will first need a lemma about approximating the square root function efficiently in parallel. While related statements are known and our statement below can be proved using standard techniques, we have included a proof in Appendix D because we do not know a reference for precisely this statement.

Lemma 13 *There is an algorithm A_r that, given an L-bit positive rational number z and an L-bit positive rational number β as input, outputs $A_r(z)$ for which $|A_r(z) - \sqrt{z}| \leq \beta$ in* poly$(\log\log(1/\beta), \log L)$ *parallel time using* poly$(\log(1/\beta), L)$ *processors.*

Armed with the ability to approximate the square root, we can easily approximate ϕ'.

Lemma 14 *There is an algorithm A_p that, given an L-bit positive rational number z, and an L-bit positive rational number $\beta \leq 1/4$, outputs $A_p(z)$ for which $|A_p(z) - \phi'(z)| \leq \beta$ in at most* poly$(\log\log(1/\beta), \log L)$ *parallel time using* poly$(\log(1/\beta), L)$ *processors.*

Proof: Assume without loss of generality that $\beta \leq 1/4$. Then, because $\sqrt{1+z^2} \geq 1$, if an approximation s of $\sqrt{1+z^2}$ satisfies $|s - \sqrt{1+z^2}| \leq \beta/2^{L+1}$, then

$$\frac{1}{s} - \frac{1}{\sqrt{1+z^2}} \leq \beta/2^L.$$

Applying Lemma 13 and recalling the well-known fact that there are efficient parallel algorithms for division (see Beame et al., 1986) completes the proof. ■

Using this approximation for ϕ', and calculating the sums in the straightforward way, we get the required approximation $\hat{\mathbf{r}}_k$. We may assume without loss of generality that each component of $\hat{\mathbf{r}}_k$ has been rounded to the nearest multiple of $\beta/2$. Since each component of g has size at most 2, and the denominator of $\hat{\mathbf{r}}_k$ has $O(K)$ bits, $\hat{\mathbf{r}}_k$ in total requires at most $O(K)$ bits. We can assume without loss of generality that $\gamma^2/50$ is a perfect square, so multiplying the components of a vector by $\frac{1-\sqrt{\mu}}{1+\sqrt{\mu}}$ can be accomplished while adding $O(\log(1/\gamma))$ bits to each of their rational representations. Thus, a straightforward induction implies that each of the components of each of the denominators of $\mathbf{v}_k$ and $\mathbf{z}_k$ can be written with $k\log(1/\gamma) + O(1/\gamma) = O((1/\gamma)\log(1/\gamma))$ bits.

To bound the numerators of the components of $\mathbf{v}_k$ and $\mathbf{z}_k$, it suffices to bound the norms of $\mathbf{v}_k$ and $\mathbf{z}_k$. Lemma 11 implies that $||\mathbf{v}_k|| \leq 5/\gamma'$ and so Lemma 6 implies $||\hat{\mathbf{v}}_k|| \leq 5/\gamma' + 1$ which in turn directly implies $||\hat{\mathbf{z}}_k|| \leq 3(5/\gamma' + 1)$.

Thus, each iteration takes $O(\text{polylog}(1/\gamma))$ time, and there are a total of $O(1/\gamma)$ iterations. This completes the proof of Theorem 1.

3. Lower Bound for Parallel Boosting in the Oracle Model

Boosting is a widely used method for learning large-margin halfspaces. In this section we consider the question of whether boosting algorithms can be efficiently parallelized. We work in the original PAC learning setting (Valiant, 1984; Kearns and Vazirani, 1994; Schapire, 1990) in which a weak learning algorithm is provided as an oracle that is called by the boosting algorithm, which must simulate a distribution over labeled examples for the weak learner. Our main result for this setting is that boosting is inherently sequential; being able to call the weak learner multiple times in parallel within a single boosting stage does not reduce the overall number of sequential boosting stages that are required. In fact we show this in a very strong sense, by proving that a boosting algorithm that runs *arbitrarily* many copies of the weak learner in parallel in each stage cannot save *even one* stage over a sequential booster that runs the weak learner just once in each stage. This lower bound is unconditional and information-theoretic.

Below we first define the parallel boosting framework and give some examples of parallel boosters. We then state and prove our lower bound on the number of stages required by parallel boosters. A consequence of our lower bound is that $\Omega(\log(1/\varepsilon)/\gamma^2)$ stages of parallel boosting are required in order to boost a γ-advantage weak learner to achieve classification accuracy $1-\varepsilon$ no matter how many copies of the weak learner are used in parallel in each stage.

3.1 Parallel Boosting

Our definition of weak learning is standard in PAC learning, except that for our discussion it suffices to consider a single target function $f : X \to \{-1,1\}$ over a domain X.

Definition 15 *A* γ-advantage weak learner *L is an algorithm that is given access to a source of independent random labeled examples drawn from an (unknown and arbitrary) probability distribution $\mathcal{P}$ over labeled examples $\{(x, f(x))\}_{x\in X}$. L must*[2] *return a weak hypothesis $h : X \to \{-1,1\}$ that satisfies* $\Pr_{(x,f(x))\leftarrow\mathcal{P}}[h(x) = f(x)] \geq 1/2+\gamma$. *Such an h is said to have* advantage γ *w.r.t. $\mathcal{P}$.*

We fix $\mathcal{P}$ to henceforth denote the initial distribution over labeled examples; that is, $\mathcal{P}$ is a distribution over $\{(x, f(x))\}_{x\in X}$ where the marginal distribution $\mathcal{P}_X$ may be an arbitrary distribution over X.

Intuitively, a boosting algorithm runs the weak learner repeatedly on a sequence of carefully chosen distributions $\mathcal{P}_1, \mathcal{P}_2, \ldots$ to obtain weak hypotheses $h_1, h_2, \ldots$, and combines the weak hypotheses to obtain a final hypothesis h that has high accuracy under $\mathcal{P}$. We first give a definition that captures the idea of a "sequential" (non-parallel) booster, and then extend the definition to parallel boosters.

3.1.1 SEQUENTIAL BOOSTERS

We give some intuition to motivate our definition. In a normal (sequential) boosting algorithm, the probability weight that the $(t+1)$st distribution $\mathcal{P}_{t+1}$ puts on a labeled example $(x, f(x))$ may depend on the values of all the previous weak hypotheses $h_1(x), \ldots, h_t(x)$ and on the value of $f(x)$. No other dependence on x is allowed, since intuitively the only interface that the boosting algorithm should have with each data point is through its label and the values of the weak hypotheses. We

2. The usual definition of a weak learner would allow L to fail with probability δ. This probability can be made exponentially small by running L multiple times so for simplicity we assume there is no failure probability.

further observe that since the distribution $\mathcal{P}$ is the only source of labeled examples, a booster should construct the distribution $\mathcal{P}_{t+1}$ by somehow "filtering" examples drawn from $\mathcal{P}$ based on the values $h_1(x),\ldots,h_t(x),f(x)$. We thus define a sequential booster as follows:

Definition 16 (Sequential booster) *A* T-stage sequential boosting algorithm *is defined by a sequence* $\alpha_1,\ldots,\alpha_T$ *of functions* $\alpha_t : \{-1,1\}^t \to [0,1]$ *and a (randomized) Boolean function* $h : \{-1,1\}^T \to \{-1,1\}$. *In the* t*-th stage of boosting, the distribution* $\mathcal{P}_t$ *over labeled examples that is given to the weak learner by the booster is obtained from* $\mathcal{P}$ *by doing rejection sampling according to* α_t. *More precisely, a draw from* $\mathcal{P}_t$ *is made as follows: draw* $(x,f(x))$ *from* $\mathcal{P}$ *and compute the value* $p_x := \alpha_t(h_1(x),\ldots,h_{t-1}(x),f(x))$. *With probability* p_x *accept* $(x,f(x))$ *as the output of the draw from* $\mathcal{P}_t$*, and with the remaining* $1-p_x$ *probability reject this* $(x,f(x))$ *and try again. In stage* t *the booster gives the weak learner access to* $\mathcal{P}_t$ *as defined above, and the weak learner generates a hypothesis* h_t *that has advantage at least* γ *w.r.t.* $\mathcal{P}_t$. *Together with* $h_1,\ldots,h_{t-1}$*, this* h_t *enables the booster to give the weak learner access to* $\mathcal{P}_{t+1}$ *in the next stage.*

After T *stages, weak hypotheses* $h_1,\ldots,h_T$ *have been obtained from the weak learner. The* final hypothesis *of the booster is* $H(x) := h(h_1(x),\ldots,h_T(x))$*, and its* accuracy *is*

$$\min_{h_1,\ldots,h_T} \Pr_{(x,f(x))\leftarrow\mathcal{P}}[H(x)=f(x)],$$

where the min is taken over all sequences $h_1,\ldots,h_T$ *of* T *weak hypotheses subject to the condition that each* h_t *has advantage at least* γ *w.r.t.* $\mathcal{P}_t$.

Many PAC-model boosting algorithms in the literature are covered by Definition 16, such as the original boosting algorithm of Schapire (1990), Boost-by-Majority (Freund, 1995), MadaBoost (Domingo and Watanabe, 2000), BrownBoost (Freund, 2001), SmoothBoost (Servedio, 2003), FilterBoost (Bradley and Schapire, 2007) and others. All these boosters use $\Omega(\log(1/\varepsilon)/\gamma^2)$ stages of boosting to achieve $1-\varepsilon$ accuracy, and indeed Freund (1995) has shown that any sequential booster must run for $\Omega(\log(1/\varepsilon)/\gamma^2)$ stages. More precisely, Freund (1995) modeled the phenomenon of boosting using the majority function to combine weak hypotheses as an interactive game between a "weightor" and a "chooser" (see Freund, 1995, Section 2). He gave a strategy for the weightor, which corresponds to a boosting algorithm, and showed that after T stages of boosting this boosting algorithm generates a final hypothesis that is guaranteed to have error at most $\mathrm{vote}(\gamma,T) \stackrel{\text{def}}{=} \sum_{j=0}^{\lfloor T/2\rfloor}\binom{T}{j}\left(\frac{1}{2}+\gamma\right)^j(1/2-\gamma)^{T-j}$ (see Freund, 1995, Theorem 2.1). Freund also gives a matching lower bound by showing (see his Theorem 2.4) that any T-stage sequential booster must have error at least as large as $\mathrm{vote}(\gamma,T)$, and so consequently any sequential booster that generates a $(1-\varepsilon)$-accurate final hypothesis must run for $\Omega(\log(1/\varepsilon)/\gamma^2)$ stages. Our Theorem 18 below extends this lower bound to parallel boosters.

3.1.2 PARALLEL BOOSTING

Parallel boosting is a natural generalization of sequential boosting. In stage t of a parallel booster the boosting algorithm may simultaneously run the weak learner many times in parallel using different probability distributions. The distributions that are used in stage t may depend on any of the weak hypotheses from earlier stages, but may not depend on any of the weak hypotheses generated by any of the calls to the weak learner in stage t.

Definition 17 (Parallel booster) *A T-stage parallel boosting algorithm with N-fold parallelism is defined by TN functions $\{\alpha_{t,k}\}_{t\in[T],k\in[N]}$ and a (randomized) Boolean function h, where $\alpha_{t,k} : \{-1,1\}^{(t-1)N+1} \to [0,1]$ and $h : \{-1,1\}^{TN} \to \{-1,1\}$. In the t-th stage of boosting the weak learner is run N times in parallel. For each $k \in [N]$, the distribution $\mathcal{P}_{t,k}$ over labeled examples that is given to the k-th run of the weak learner is as follows: a draw from $\mathcal{P}_{t,k}$ is made by drawing a labeled example $(x, f(x))$ from $\mathcal{P}$, computing the value $p_x := \alpha_{t,k}(h_{1,1}(x), \ldots, h_{t-1,N}(x), f(x))$, and accepting $(x, f(x))$ as the output of the draw from $\mathcal{P}_{t,k}$ with probability p_x (and rejecting it and trying again otherwise). In stage t, for each $k \in [N]$ the booster gives the weak learner access to $\mathcal{P}_{t,k}$ as defined above and the weak learner generates a hypothesis $h_{t,k}$ that has advantage at least γ w.r.t. $\mathcal{P}_{t,k}$. Together with the weak hypotheses $\{h_{s,j}\}_{s\in[t-1],j\in[N]}$ obtained in earlier stages, these $h_{t,k}$'s enable the booster to give the weak learner access to each $\mathcal{P}_{t+1,k}$ in the next stage.*

After T stages, TN weak hypotheses $\{h_{t,k}\}_{t\in[T],k\in[N]}$ have been obtained from the weak learner. The final hypothesis *of the booster is $H(x) := h(h_{1,1}(x), \ldots, h_{T,N}(x))$, and its* accuracy *is*

$$\min_{h_{t,k}} \Pr_{(x,f(x))\leftarrow\mathcal{P}}[H(x) = f(x)],$$

where the min is taken over all sequences of TN weak hypotheses subject to the condition that each $h_{t,k}$ has advantage at least γ w.r.t. $\mathcal{P}_{t,k}$.

The parameter N above corresponds to the number of processors that the parallel booster is using. Parallel boosting algorithms that call the weak learner different numbers of times at different stages fit into our definition simply by taking N to be the max number of parallel calls made at any stage. Several parallel boosting algorithms have been given in the literature; in particular, all boosters that construct branching program or decision tree hypotheses are of this type. The number of stages of these boosting algorithms corresponds to the depth of the branching program or decision tree that is constructed, and the number of nodes at each depth corresponds to the parallelism parameter. Branching program boosters (Mansour and McAllester, 2002; Kalai and Servedio, 2005; Long and Servedio, 2005, 2008) all make $\mathrm{poly}(1/\gamma)$ many calls to the weak learner within each stage and all require $\Omega(\log(1/\varepsilon)/\gamma^2)$ stages, while the earlier decision tree booster (Kearns and Mansour, 1996) requires $\Omega(\log(1/\varepsilon)/\gamma^2)$ stages but makes $2^{\Omega(\log(1/\varepsilon)/\gamma^2)}$ parallel calls to the weak learner in some stages. Our results in the next subsection will imply that *any* parallel booster must run for $\Omega(\log(1/\varepsilon)/\gamma^2)$ stages no matter how many parallel calls to the weak learner are made in each stage.

3.2 The Lower Bound and Its Proof

Our lower bound theorem for parallel boosting is the following:

Theorem 18 *Let B be any T-stage parallel boosting algorithm with N-fold parallelism. Then for any $0 < \gamma < 1/2$, when B is used to boost a γ-advantage weak learner the resulting final hypothesis may have error as large as* $\mathrm{vote}(\gamma, T)$ *(see the discussion after Definition 17).*

We emphasize that Theorem 18 holds for any γ and any N that may depend on γ in an arbitrary way.

The theorem is proved as follows: fix any $0 < \gamma < 1/2$ and fix B to be any T-stage parallel boosting algorithm. We will exhibit a target function f and a distribution $\mathcal{P}$ over $\{(x, f(x))_{x\in X}$, and

describe a strategy that a weak learner W can use to generate weak hypotheses $h_{t,k}$ that all have advantage at least γ with respect to the distributions $\mathcal{P}_{t,k}$. We show that with this weak learner W, the resulting final hypothesis H that B outputs will have accuracy at most $1 - \text{vote}(\gamma, T)$.

We begin by describing the desired f and $\mathcal{P}$, both of which are fairly simple. The domain X of f is $X = Z \times \Omega$, where Z denotes the set $\{-1,1\}$ and Ω denotes the set of all infinite sequences $\omega = (\omega_1, \omega_2, \ldots)$ where each ω_i belongs to $\{-1,1\}$. The target function f is simply $f(z,\omega) = z$; that is, f always simply outputs the first coordinate of its input vector. The distribution $\mathcal{P} = (\mathcal{P}^X, \mathcal{P}^Y)$ over labeled examples $\{(x, f(x))\}_{x \in X}$ is defined as follows.[3] A draw from $\mathcal{P}$ is obtained by drawing $x = (z,\omega)$ from $\mathcal{P}^X$ and returning $(x, f(x))$. A draw of $x = (z,\omega)$ from $\mathcal{P}^X$ is obtained by first choosing a uniform random value in $\{-1,1\}$ for z, and then choosing $\omega_i \in \{-1,1\}$ to equal z with probability $1/2 + \gamma$ independently for each i. Note that under $\mathcal{P}$, given the label $z = f(x)$ of a labeled example $(x, f(x))$, each coordinate ω_i of x is correct in predicting the value of $f(x,z)$ with probability $1/2 + \gamma$ independently of all other ω_j's.

We next describe a way that a weak learner W can generate a γ-advantage weak hypothesis each time it is invoked by B. Fix any $t \in [T]$ and any $k \in [N]$, and recall that $\mathcal{P}_{t,k}$ is the distribution over labeled examples that is used for the k-th call to the weak learner in stage t. When W is invoked with $\mathcal{P}_{t,k}$ it replies as follows (recall that for $x \in X$ we have $x = (z,\omega)$ as described above):

(i) If $\Pr_{(x,f(x)) \leftarrow \mathcal{P}_{t,k}}[\omega_t = f(x)] \geq 1/2 + \gamma$ then the weak hypothesis $h_{t,k}(x)$ is the function "ω_t," the $(t+1)$-st coordinate of x. Otherwise,

(ii) the weak hypothesis $h_{t,k}(x)$ is "z," the first coordinate of x. (Note that since $f(x) = z$ for all x, this weak hypothesis has zero error.)

It is clear that each weak hypothesis $h_{t,k}$ generated as described above indeed has advantage at least γ w.r.t. $\mathcal{P}_{t,k}$, so the above is a legitimate strategy for W. It is also clear that if the weak learner ever uses option (ii) above at some invocation (t,k) then B may output a zero-error final hypothesis simply by taking $H = h_{t,k} = f(x)$. On the other hand, the following crucial lemma shows that if the weak learner never uses option (ii) for any (t,k) then the accuracy of B is upper bounded by $\text{vote}(\gamma, T)$:

Lemma 19 *If W never uses option (ii) then* $\Pr_{(x,f(x)) \leftarrow \mathcal{P}}[H(x) \neq f(x)] \geq \text{vote}(\gamma, T)$.

Proof If the weak learner never uses option (ii) then H depends only on variables

$$\omega_1, \ldots, \omega_T$$

and hence is a (randomized) Boolean function over these variables. Recall that for $(x = (z,\omega), f(x) = z)$ drawn from $\mathcal{P}$, each coordinate

$$\omega_1, \ldots, \omega_T$$

independently equals z with probability $1/2 + \gamma$. Hence the optimal (randomized) Boolean function H over inputs $\omega_1, \ldots, \omega_T$ that maximizes the accuracy $\Pr_{(x,f(x)) \leftarrow \mathcal{P}}[H(x) = f(x)]$ is the (deterministic) function $H(x) = \mathsf{Maj}(\omega_1, \ldots, \omega_T)$ that outputs the majority vote of its input bits. (This can be

3. Note that $\mathcal{P}^X$ and $\mathcal{P}^Y$ are not independent; indeed, in a draw $(x, y = f(x))$ from $(\mathcal{P}^X, \mathcal{P}^Y)$ the outcome of x completely determines y.

easily verified using Bayes' rule in the usual "Naive Bayes" calculation.) The error rate of this H is precisely the probability that at most $\lfloor T/2 \rfloor$ "heads" are obtained in T independent $(1/2+\gamma)$-biased coin tosses, which equals $\text{vote}(\gamma, T)$. ■

Thus to prove Theorem 18 it suffices to prove the following lemma, which we prove by induction on t:

Lemma 20 *W never uses option (ii) (that is,* $\Pr_{(x,f(x))\leftarrow\mathcal{P}_{t,k}}[\omega_t = f(x)] \geq 1/2+\gamma$ *always).*

Proof *Base case ($t = 1$).* For any $k \in [N]$, since $t = 1$ there are no weak hypotheses from previous stages, so the value of the rejection sampling parameter p_x is determined by the bit $f(x) = z$ (see Definition 17). Hence $\mathcal{P}_{1,k}$ is a convex combination of two distributions which we call $\mathcal{D}_1$ and $\mathcal{D}_{-1}$. For $b \in \{-1,1\}$, a draw of $(x = (z,\omega); f(x) = z)$ from $\mathcal{D}_b$ is obtained by setting $z = b$ and independently setting each coordinate ω_i equal to z with probability $1/2+\gamma$. Thus in the convex combination $\mathcal{P}_{1,k}$ of $\mathcal{D}_1$ and $\mathcal{D}_{-1}$, we also have that ω_1 equals z (that is, $f(x)$) with probability $1/2+\gamma$. So the base case is done.

Inductive step ($t > 1$). Thanks to the conditional independence of different coordinates ω_i given the value of z in a draw from $\mathcal{P}$, the proof is quite similar to the base case.

Fix any $k \in [N]$. The inductive hypothesis and the weak learner's strategy together imply that for each labeled example $(x = (z,\omega), f(x) = z)$, since $h_{s,\ell}(x) = \omega_s$ for $s < t$, the rejection sampling parameter $p_x = \alpha_{t,k}(h_{1,1}(x), \ldots, h_{t-1,N}(x), f(x))$ is determined by $\omega_1, \ldots, \omega_{t-1}$ and z and does not depend on $\omega_t, \omega_{t+1}, \ldots$. Consequently the distribution $\mathcal{P}_{t,k}$ over labeled examples is some convex combination of 2^t distributions which we denote $\mathcal{D}_{\overline{b}}$, where $\overline{b}$ ranges over $\{-1,1\}^t$ corresponding to conditioning on all possible values for $\omega_1, \ldots, \omega_{t-1}, z$. For each $\overline{b} = (b_1, \ldots, b_t) \in \{-1,1\}^t$, a draw of $(x = (z,\omega); f(x) = z)$ from $\mathcal{D}_{\overline{b}}$ is obtained by setting $z = b_t$, setting $(\omega_1, \ldots, \omega_{t-1}) = (b_1, \ldots, b_{t-1})$, and independently setting each other coordinate ω_j $(j \geq t)$ equal to z with probability $1/2+\gamma$. In particular, because ω_t is conditionally independent of $\omega_1, \ldots, \omega_{t-1}$ given z, $\Pr(\omega_t = z | \omega_1 = b_1, \ldots, \omega_{t-1} = b_{t-1}) = \Pr(\omega_t = z) = 1/2+\gamma$. Thus in the convex combination $\mathcal{P}_{t,k}$ of the different $\mathcal{D}_{\overline{b}}$'s, we also have that ω_t equals z (that is, $f(x)$) with probability $1/2+\gamma$. This concludes the proof of the lemma and the proof of Theorem 18. ■

4. Conclusion

There are many natural directions for future work on understanding the parallel complexity of learning large-margin halfspaces. One natural goal, of course, is to give an algorithm that provides an affirmative answer to the main question. But it is not clear to us that such an algorithm must actually exist, and so another intriguing direction is to prove negative results giving evidence that parallel learning of large-margin halfspaces is computationally hard.

As one example of a possible negative result, perhaps it is the case that (assuming $P \neq NC$) there is no $\text{poly}(n)$-processor, $\text{polylog}(n)$-time algorithm with the following performance guarantee: given a sample of $\text{poly}(n)$ many n-dimensional labeled examples that are consistent with some $1/\text{poly}(n)$-margin halfspace, the algorithm outputs a consistent halfspace hypothesis. A stronger result would be that no such algorithm can even output a halfspace hypothesis which is consistent

with 99% (or 51%) of the labeled examples. Because of the requirement of a halfspace representation for the hypothesis such results would not directly contradict the main question, but they are contrary to it in spirit. We view the possibility of establishing such negative results as an interesting direction worthy of future study.

Acknowledgments

We thank Sasha Rakhlin for telling us about the paper of Soheili and Peña (2012), and anonymous reviewers for helpful comments.

Appendix A. Proof of Lemma 2

First, let us establish that we can "boost the confidence" efficiently. Suppose we have an algorithm that achieves accuracy $1-\varepsilon$ in parallel time $\mathcal{T}''$ with probability c_δ. Then we can run $O(\log(1/\delta))$ copies of this algorithm in parallel, then test each of their hypotheses in parallel using $O(\log(1/\delta)/\varepsilon)$ examples. The tests of individual examples can be done in parallel, and we can compute each empirical error rate in $O(\log(1/\varepsilon)+\log\log(1/\delta))$ time. Then we can output the hypothesis with the best accuracy on this additional test data. Finding the best hypothesis takes at most $O(\log\log(1/\delta))$ parallel time (with polynomially many processors). The total parallel time taken is then $O(\mathcal{T}''+\log(1/\varepsilon)+\log\log(1/\delta))$.

So now, we have as a subproblem the problem of achieving accuracy $1-\varepsilon$ with constant probability, say $1/2$.

The theorem statement assumes that we have as a subroutine an algorithm A that achieves constant accuracy with constant probability in time $\mathcal{T}$. Using the above reduction, we can use A to get an algorithm A' that achieves constant accuracy with probability $1-c/\log(1/\varepsilon)$ (for a constant c) in $\mathcal{T}'=O(\mathcal{T}+\log\log\log(1/\varepsilon))$ time. We will use such an algorithm A'. (Note that the time taken by A' is an upper bound on the number of examples needed by A'.)

Algorithm B runs a parallel version of a slight variant of the "boosting-by-filtering" algorithm due to Freund (1995), using A' as a weak learner. Algorithm B uses parameters α and T:

- For rounds $t=0,\ldots,T-1$
  - draw $m=\frac{2T\alpha}{\varepsilon}\max\{\mathcal{T}',4\ln\frac{32T^2\alpha}{\varepsilon}\}$ examples, call them
    $$S_t=\{(x_{t,1},y_{t,1}),\ldots,(x_{t,m},y_{t,m})\}.$$
  - for each $i=1,\ldots,m$,
    - let $r_{t,i}$ be the the number of previous base classifiers h_0,...,h_{t-1} that are correct on $(x_{t,i},y_{t,i})$, and
    - $w_{t,i}=\binom{T-t-1}{\lfloor\frac{T}{2}\rfloor-r_{t,i}}(\frac{1}{2}+\alpha)^{\lfloor\frac{T}{2}\rfloor-r_{t,i}}(\frac{1}{2}-\alpha)^{\lceil\frac{T}{2}\rceil-t-1+r_{t,i}}$,
  - let $w_{t,\max}=\max_r\binom{T-t-1}{\lfloor\frac{T}{2}\rfloor-r}(\frac{1}{2}+\alpha)^{\lfloor\frac{T}{2}\rfloor-r}(\frac{1}{2}-\alpha)^{\lceil\frac{T}{2}\rceil-t-1+r}$ be the largest possible value that any $w_{t,i}$ could take,
  - apply the rejection method as follows: for each $i\in S_t$,
    - choose $u_{t,i}$ uniformly from $[0,1]$,

  - ∗ if $u_{t,i} \leq \frac{w_{t,i}}{w_{t,\max}}$, set $a_{t,i} = 1$
- if there is a j such that $j > \frac{T\alpha w_{t,\max}}{\varepsilon} \max\left\{\sum_{i=1}^{j} a_{t,i}, 4\ln\frac{16T^2\alpha w_{t,\max}}{\varepsilon(1-\varepsilon)}\right\}$
  - ∗ output a hypothesis h_t that predicts randomly,
  - ∗ otherwise, pass the examples in S_t to Algorithm A', which returns h_t.

- Output the classifier obtained by taking a majority vote over $h_0, ..., h_{T-1}$.

The only difference between algorithm B, as described above, and the way the algorithm is described by Freund (1995) is that, in the above description, a batch of examples is chosen at the beginning of the round. The number of examples is set using Freund's upper bound on the number of examples that can be chosen in a given round (see the displayed equation of the boost-by-majority paper (Freund, 1995) immediately before (18)). In Freund's description of this algorithm, once the condition which causes the algorithm to output a random hypothesis is reached, the algorithm stops sampling, but, for a parallel version, it is convenient to sample all of the examples for a round in parallel.

Freund (1995) proves that, if α is a constant depending only on the accuracy of the hypotheses output by A', then $T = O(\log(1/\varepsilon))$ suffices for algorithm B to output a hypothesis with accuracy $1-\varepsilon$ with probability $1/2$. So the parallel time taken is $O(\log(1/\varepsilon))$ times the time taken in each iteration.

Let us now consider the time taken in each iteration. The weights for the various examples can be computed in parallel. The value of $w_{t,i}$ is a product of $O(T)$ quantities, each of which can be expressed using T bits, and can therefore be computed in $O(\text{poly}(\log T)) = O(\text{poly}(\log\log(1/\varepsilon)))$ parallel time, as can $w_{t,\max}$. The rejection step also may be done in $O(\text{poly}(\log\log(1/\varepsilon)))$ time in parallel for each example. To check whether there is a j such that

$$j > \frac{T\alpha w_{t,\max}}{\varepsilon} \max\left\{\sum_{i=1}^{j} a_{t,i}, 4\ln\frac{16T^2\alpha w_{t,\max}}{\varepsilon(1-\varepsilon)}\right\},$$

Algorithm B can compute the prefix sums $\sum_{i=1}^{j} a_{t,i}$, and then test them in parallel. The prefix sums can be computed in $\log(T)$ parallel rounds (each on $\log(T)$-bit numbers), using the standard technique of placing the values of $a_{t,i}$ on the leaves of a binary tree, and working up from the leaves to the root, computing the sums of subtrees, then making a pass down the tree, passing each node's sum to its right child, and using these to compute prefix sums in the obvious way.

Appendix B. Proof of Lemma 5

Algorithm A_{Nes} is a special case of the algorithm of (2.2.11) on page 81 of the book by Nesterov (2004), obtained by setting $y_0 \leftarrow \mathbf{0}$ and $x_0 \leftarrow \mathbf{0}$. The bound of Lemma 5 is a consequence of Theorem 2.2.3 on page 80 of Nesterov's book. This Theorem applies to all functions f that are μ-strongly convex, and continuously differentiable with a gradient that is L-Lipschitz (see pages 71, 63 and 20). Lemmas 3 and 4 of this paper imply that Theorem 2.2.3 of Nesterov's book applies to Ψ.

Plugging directly into Theorem 2.2.3 (in the special case of (2.2.11))

$$\Psi(\mathbf{v}_k) - \Psi(\mathbf{w}) \leq \frac{4L}{(2\sqrt{L} + k\sqrt{\mu})^2}(\Psi(\mathbf{0}) - \Psi(\mathbf{w}) + \mu||\mathbf{w}||^2)$$

which implies the Lemma 5, since $\Psi(\mathbf{0}) \leq 1$ and $\Psi(\mathbf{w}) \geq 0$.

Appendix C. Proof of Lemma 7

First, we prove

$$\Pr_A[\Pr_{\mathbf{x}' \sim \mathcal{D}'}\left[||\mathbf{x}'|| > 2\right] > \gamma^4/2] \; < \; 1/200. \tag{2}$$

Recall that we sample $\mathbf{x}'$ from D' by first sampling $\mathbf{x}$ from a distribution D over B_n (so that $||\mathbf{x}|| = 1$), and then setting $\mathbf{x}' = (1/\sqrt{d})\mathbf{x}A$, so that (2) is equivalent to

$$\Pr_A[\Pr_{\mathbf{x} \sim \mathcal{D}}\left[||\mathbf{x}A|| > 2\sqrt{d}\right] > \gamma^4/2] \; < \; 1/200.$$

Corollary 1 of the paper of Arriaga and Vempala (2006) directly implies that, for any $\mathbf{x}$ in $\mathbf{B}_n$, we have

$$\Pr_A[||\mathbf{x}A|| \geq 2\sqrt{d}] \leq 2e^{-\frac{d}{32}},$$

so

$$\mathbf{E}_{\mathbf{x} \in D}[\Pr_A[||\mathbf{x}A|| \geq 2\sqrt{d}]] \leq 2e^{-\frac{d}{32}},$$

which implies

$$\mathbf{E}_A[\Pr_{\mathbf{x} \in D}[||\mathbf{x}A|| \geq 2\sqrt{d}]] \leq 2e^{-\frac{d}{32}}.$$

Applying Markov's inequality,

$$\Pr_A[\Pr_{\mathbf{x} \in D}[||\mathbf{x}A|| \geq 2\sqrt{d}] > 400e^{-\frac{d}{32}}] \leq 1/200.$$

Setting $d = O(\log(1/\gamma))$ then suffices to establish (2).

Now, we want to show that $d = O((1/\gamma^2)\log(1/\gamma))$ suffices to ensure that

$$\Pr_A\left[\Pr_{\mathbf{x}' \sim \mathcal{D}'}\left[\left|\frac{\mathbf{w}'}{||\mathbf{w}'||} \cdot \mathbf{x}'\right| < \gamma/2\right] > \gamma^4/2\right] \leq 1/200.$$

As above, Corollary 1 of the paper by Arriaga and Vempala (2006) directly implies that there is an absolute constant $c_1 > 0$ such that

$$\Pr_A[||\mathbf{w}'|| = ||(1/\sqrt{d})\mathbf{w}A|| > 3/2] \leq 2e^{-c_1 d}.$$

Furthermore, for any $\mathbf{x} \in \mathbf{B}_n$, Corollary 2 of the paper by Arriaga and Vempala (2006) directly implies that there is an absolute constant $c_2 > 0$ such that

$$\Pr_A[\mathbf{w}' \cdot \mathbf{x}' \leq 3\gamma/4] \leq 4e^{-c_2\gamma^2 d}.$$

Thus,

$$\Pr_A\left[\frac{\mathbf{w}'}{||\mathbf{w}'||} \cdot \mathbf{x}' \leq \gamma/2\right] \leq 2e^{-c_1 d} + 4e^{-c_2\gamma^2 d}.$$

Arguing as above, we have

$$\mathbf{E}_{\mathbf{x} \in D}\left[\Pr_A\left[\frac{\mathbf{w}'}{||\mathbf{w}'||} \cdot \mathbf{x}' \leq \gamma/2\right]\right] \leq 2e^{-c_1 d} + 4e^{-c_2\gamma^2 d},$$

$$\mathbf{E}_A\left[\Pr_{\mathbf{x} \in D}\left[\frac{\mathbf{w}'}{||\mathbf{w}'||} \cdot \mathbf{x}' \leq \gamma/2\right]\right] \leq 2e^{-c_1 d} + 4e^{-c_2\gamma^2 d},$$

$$\Pr_A\left[\Pr_{\mathbf{x} \in D}\left[\frac{\mathbf{w}'}{||\mathbf{w}'||} \cdot \mathbf{x}' \leq \gamma/2\right] > 200(2e^{-c_1 d} + 4e^{-c_2\gamma^2 d})\right] \leq 1/200,$$

from which $d = O((1/\gamma^2)\log(1/\gamma))$ suffices to get

$$\Pr_A\left[\Pr_{\mathbf{x}\in D}\left[\frac{\mathbf{w}'}{||\mathbf{w}'||}\cdot\mathbf{x}' \leq \gamma/2\right] > \gamma^A/2\right] \leq 1/200,$$

completing the proof.

Appendix D. Proof of Lemma 13

First, A_r finds a rough guess u_1 such that

$$\sqrt{z}/2 \leq u_1 \leq \sqrt{z}. \tag{3}$$

This can be done by checking in parallel, for each of $\theta \in \{1/2^L, 1/2^{L-1}, ..., 1/2, 1, 2, ..., 2^L\}$, whether $\sqrt{z} \geq \theta$, and outputting the largest such θ. This first step takes $O(\log L)$ time using $O(L)$ processors. Then, using u_1 as the initial solution, A_r runs Newton's method to find a root of the function f defined by $f(u) = u^2 - z$, repeatedly

$$u_{k+1} = \frac{1}{2}\left(u_k + \frac{z}{u_k}\right). \tag{4}$$

As we will see below, this is done for $k = 1, \ldots, O(\log L + \log\log(1/\beta))$. Using the fact that the initial value u_1 is an L-bit rational number, a straightforward analysis using (4) shows that for all $k \leq O(\log L + \log\log(1/\beta))$ the number u_k is a rational number with $\text{poly}(L, \log(1/\beta))$ bits (if b_k is the number of bits required to represent u_k, then $b_{k+1} \leq 2b_k + O(L)$). Standard results on the parallel complexity of integer multiplication thus imply that for $k \leq O(\log L + \log\log(1/\beta))$ the exact value of u_k can be computed in the parallel time and processor bounds claimed by the Lemma. To prove the Lemma, then, it suffices to show that taking $k = O(\log L + \log\log(1/\beta))$ gives the desired accuracy; we do this next.

The Newton iterates defined by (4) satisfy

$$\frac{u_{k+1} - \sqrt{z}}{u_{k+1} + \sqrt{z}} = \left(\frac{u_k - \sqrt{z}}{u_k + \sqrt{z}}\right)^2$$

(see Weisstein, 2011), which, using induction, gives

$$\frac{u_{k+1} - \sqrt{z}}{u_{k+1} + \sqrt{z}} = \left(\frac{u_1 - \sqrt{z}}{u_1 + \sqrt{z}}\right)^{2^k}.$$

Solving for u_{k+1} yields

$$u_{k+1} = \sqrt{z}\left(\frac{1 + \left(\frac{u_1-\sqrt{z}}{u_1+\sqrt{z}}\right)^{2^k}}{1 - \left(\frac{u_1-\sqrt{z}}{u_1+\sqrt{z}}\right)^{2^k}}\right) = \sqrt{z}\left(1 + \frac{2\left(\frac{u_1-\sqrt{z}}{u_1+\sqrt{z}}\right)^{2^k}}{1 - \left(\frac{u_1-\sqrt{z}}{u_1+\sqrt{z}}\right)^{2^k}}\right).$$

Thus,

$$u_{k+1} - \sqrt{z} = \frac{2\sqrt{z}\left(\frac{u_1-\sqrt{z}}{u_1+\sqrt{z}}\right)^{2^k}}{1 - \left(\frac{u_1-\sqrt{z}}{u_1+\sqrt{z}}\right)^{2^k}}$$

and, therefore, to get $|u_{k+1} - \sqrt{z}| \leq \beta$, we only need

$$\left(\frac{u_1 - \sqrt{z}}{u_1 + \sqrt{z}}\right)^{2^k} \leq \min\left\{\frac{\beta}{4\sqrt{z}}, 1/2\right\}.$$

Applying (3),

$$(1/4)^{2^k} \leq \min\left\{\frac{\beta}{4\sqrt{z}}, 1/2\right\}$$

also suffices, and, solving for k, this means that

$$O(\log\log z + \log\log(1/\beta)) = O(\log L + \log\log(1/\beta))$$

iterations are enough. ■

References

N. Alon and N. Megiddo. Parallel linear programming in fixed dimension almost surely in constant time. *J. ACM*, 41(2):422–434, 1994.

R. I. Arriaga and Santosh Vempala. An algorithmic theory of learning: Robust concepts and random projection. *Machine Learning*, 63(2):161–182, 2006.

P. Beame, S.A. Cook, and H.J. Hoover. Log depth circuits for division and related problems. *SIAM J. on Computing*, 15(4):994–1003, 1986.

H. Block. The Perceptron: A model for brain functioning. *Reviews of Modern Physics*, 34:123–135, 1962.

A. Blum. Random Projection, Margins, Kernels, and Feature-Selection. In *Subspace, Latent Structure and Feature Selection*, pages 52–68, 2006.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

J. Bradley, A. Kyrola, D. Bickson, and C. Guestrin. Parallel coordinate descent for l1-regularized loss minimization. In *Proc. 28th ICML*, pages 321–328, 2011.

J. K. Bradley and R. E. Schapire. Filterboost: Regression and classification on large datasets. In *Proc. 21st NIPS*, 2007.

N. Bshouty, S. Goldman, and H.D. Mathias. Noise-tolerant parallel learning of geometric concepts. *Inf. and Comput.*, 147(1):89–110, 1998. ISSN 0890-5401. doi: DOI: 10.1006/inco.1998.2737.

M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, adaboost and bregman distances. *Machine Learning*, 48(1-3):253–285, 2002.

A. d'Aspremont. Smooth optimization with approximate gradient. *SIAM Journal on Optimization*, 19 (3): 1171–1183, 2008.

O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao. Optimal distributed online prediction. In *Proc. 28th ICML*, pages 713–720, 2011.

C. Domingo and O. Watanabe. MadaBoost: A modified version of AdaBoost. In *Proc. 13th COLT*, pages 180–189, 2000.

Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121 (2): 256–285, 1995.

Y. Freund. An adaptive version of the boost-by-majority algorithm. *Machine Learning*, 43(3):293–318, 2001.

Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

R. Greenlaw, H.J. Hoover, and W.L. Ruzzo. *Limits to Parallel Computation: P-Completeness Theory*. Oxford University Press, New York, 1995.

A. Kalai and R. Servedio. Boosting in the presence of noise. *Journal of Computer & System Sciences*, 71(3):266–290, 2005.

N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinat.*, 4:373–395, 1984.

M. Kearns and Y. Mansour. On the boosting ability of top-down decision tree learning algorithms. In *Proc. 28th STOC*, pages 459–468, 1996.

M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA, 1994.

N. Littlestone. From online to batch learning. In *Proc. 2nd COLT*, pages 269–284, 1989.

P. Long and R. Servedio. Martingale boosting. In *Proc. 18th COLT*, pages 79–94, 2005.

P. Long and R. Servedio. Adaptive martingale boosting. In *Proc. 22nd NIPS*, pages 977–984, 2008.

P. Long and R. Servedio. Algorithms and hardness results for parallel large margin learning. In *Proc. 25th NIPS*, 2011.

Y. Mansour and D. McAllester. Boosting using branching programs. *Journal of Computer & System Sciences*, 64(1):103–112, 2002.

Y. Nesterov. *Introductory lectures on Convex Optimization*. Kluwer, 2004.

Y. Nesterov. Excessive gap technique in nonsmooth convex minimization. *SIAM J. Optimization*, 16(1):235–249, 2005.

A. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on Mathematical Theory of Automata*, volume XII, pages 615–622, 1962.

F. Rosenblatt. The Perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

R. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

R. Servedio. Smooth boosting and learning with malicious noise. *JMLR*, 4:633–648, 2003.

S. Shalev-Shwartz and Y. Singer. On the equivalence of weak learnability and linear separability: New relaxations and efficient boosting algorithms. *Machine Learning*, 80(2):141–163, 2010.

N. Soheili and J. Peña. A smooth perceptron algorithm. *SIAM J. Optimization*, 22(2):728–737, 2012.

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

V. N. Vapnik and A. Y. Chervonenkis. *Theory of Pattern Recognition*. Nauka, 1974. In Russian.

J. S. Vitter and J. Lin. Learning in parallel. *Inf. Comput.*, 96(2):179–202, 1992.

E. W. Weisstein. Newton's iteration, 2011. http://mathworld.wolfram.com/NewtonsIteration.html.

DIMACS 2011 Workshop. Parallelism: A 2020 Vision. 2011.

NIPS 2009 Workshop. Large-Scale Machine Learning: Parallelism and Massive Datasets. 2009.

Journal of Machine Learning Research 14 (2013) 3129-3152 Submitted 5/12; Revised 5/13; Published 10/13

Large-scale SVD and Manifold Learning

Ameet Talwalkar AMEET@CS.BERKELEY.EDU
University of California, Berkeley
Division of Computer Science
465 Soda Hall
Berkeley, CA 94720

Sanjiv Kumar SANJIVK@GOOGLE.COM
Google Research
76 Ninth Avenue
New York, NY 10011

Mehryar Mohri MOHRI@CS.NYU.EDU
Courant Institute and Google Research
251 Mercer Street
New York, NY 10012

Henry Rowley HAR@GOOGLE.COM
Google Research
Amphitheatre Parkway
Mountain View, CA 94043

Editor: Inderjit Dhillon

Abstract

This paper examines the efficacy of sampling-based low-rank approximation techniques when applied to large dense kernel matrices. We analyze two common approximate singular value decomposition techniques, namely the Nyström and Column sampling methods. We present a theoretical comparison between these two methods, provide novel insights regarding their suitability for various tasks and present experimental results that support our theory. Our results illustrate the relative strengths of each method. We next examine the performance of these two techniques on the large-scale task of extracting low-dimensional manifold structure given millions of high-dimensional face images. We address the computational challenges of non-linear dimensionality reduction via Isomap and Laplacian Eigenmaps, using a graph containing about 18 million nodes and 65 million edges. We present extensive experiments on learning low-dimensional embeddings for two large face data sets: CMU-PIE (35 thousand faces) and a web data set (18 million faces). Our comparisons show that the Nyström approximation is superior to the Column sampling method for this task. Furthermore, approximate Isomap tends to perform better than Laplacian Eigenmaps on both clustering and classification with the labeled CMU-PIE data set.

Keywords: low-rank approximation, manifold learning, large-scale matrix factorization

1. Introduction

Kernel-based algorithms (Schölkopf and Smola, 2002) are a broad class of learning algorithms with rich theoretical underpinnings and state-of-the-art empirical performance for a variety of problems, for example, Support Vector Machines (SVMs) and Kernel Logistic Regression (KLR) for classifi-

cation, Support Vector Regression (SVR) and Kernel Ridge Regression (KRR) for regression, Kernel Principle Component Analysis (KPCA) for non-linear dimensionality reduction, SVM-Rank for ranking, etc. Despite the favorable properties of kernel methods in terms of theory, empirical performance and flexibility, scalability remains a major drawback. Given a set of n datapoints, these algorithms require $O(n^2)$ space to store the kernel matrix. Furthermore, they often require $O(n^3)$ time, requiring matrix inversion, Singular Value Decomposition (SVD) or quadratic programming in the case of SVMs. For large-scale data sets, both the space and time requirements quickly become intractable. Various optimization methods have been introduced to speed up kernel methods, for example, SMO (Platt, 1999), shrinking (Joachims, 1999), chunking (Boser et al., 1992), parallelized SVMs (Chang et al., 2008) and parallelized KLR (Mann et al., 2009). However for large-scale problems, the storage and processing costs can nonetheless be intractable.

In this work,[1] we focus on an attractive solution to this problem that involves efficiently generating low-rank approximations to the kernel matrix. Low-rank approximation appears in a wide variety of applications including lossy data compression, image processing, text analysis and cryptography, and is at the core of widely used algorithms such as Principle Component Analysis, Multidimensional Scaling and Latent Semantic Indexing. Moreover, kernel matrices can often be well approximated by low-rank matrices, the latter of which are much easier to store and operate on. Although SVD can be used to find 'optimal' low-rank approximations, SVD requires storage of the full kernel matrix and the runtime is superlinear in n, and hence does not scale well for large-scale applications.

For matrices of special form such as tridiagonal matrices, fast parallelized decomposition algorithms exist (Dhillon and Parlett, 2004), but such methods are not generally applicable. Kernel functions are sometimes chosen to yield sparse kernel matrices. When dealing with these sparse matrices, iterative methods such as the Jacobi or Arnoldi techniques (Golub and Loan, 1983) can be used to compute a compact SVD. More recent methods based on random projections (Halko et al., 2009) and statistical leverage scores (Mahoney, 2011) can yield high-quality approximations more efficiently than these standard iterative methods for sparse matrices. However, all of these methods require operating on the full matrix, and in applications where the associated kernel matrices are dense, working with full kernel matrix can be intractable. For instance, given a data set of 18M data points, as in application presented in this work, storing a dense kernel matrix would require 1300TB, and even if we could somehow store it, performing $O(n^2)$ operations would be infeasible.

When working with large dense matrices, sampling-based techniques provide a powerful alternative, as they construct low-rank matrices that are nearly 'optimal' while also having linear space and time constraints with respect to n. In this work, we focus on two commonly used sampling-based techniques, the Nyström method (Williams and Seeger, 2000) and the Column sampling method (Frieze et al., 1998). The Nyström approximation has been studied in the machine learning community (Williams and Seeger, 2000; Drineas and Mahoney, 2005), while Column sampling techniques have been analyzed in the theoretical Computer Science community (Frieze et al., 1998; Drineas et al., 2006; Deshpande et al., 2006). However, the relationship between these approximations had not been well studied. Here we provide an extensive theoretical analysis of these algorithms, show connections between these approximations and provide a direct comparison between their performances.

1. Portions of this work have previously appeared in preliminary forms in the Conference on Vision and Pattern Recognition (Talwalkar et al., 2008) and the International Conference on Machine Learning (Kumar et al., 2009).

We then examine the performance of these two low-rank approximation techniques on the task of extracting low-dimensional manifold structure given millions of high-dimensional face images. The problem of dimensionality reduction arises in many computer vision applications where it is natural to represent images as vectors in a high-dimensional space. Manifold learning techniques extract low-dimensional structure from high-dimensional data in an unsupervised manner. This makes certain applications such as K-means clustering more effective in the transformed space. Instead of assuming global linearity as in the case of linear dimensionality reduction techniques, manifold learning methods typically make a weaker local-linearity assumption, that is, for nearby points in high-dimensional input space, l_2 distance is assumed to be a good measure of geodesic distance, or distance along the manifold. Good sampling of the underlying manifold is essential for this assumption to hold. In fact, many manifold learning techniques provide guarantees that the accuracy of the recovered manifold increases as the number of data samples increases (Tenenbaum et al., 2000; Donoho and Grimes, 2003). However, there is a trade-off between improved sampling of the manifold and the computational cost of manifold learning algorithms, and we explore these computational challenges in this work.

We focus on Isomap (Tenenbaum et al., 2000) and Laplacian Eigenmaps (Belkin and Niyogi, 2001), as both methods have good theoretical properties and the differences in their approaches allow us to make interesting comparisons between dense and sparse methods. Isomap in particular involves storing and operating on a dense similarity matrix, and although the similarity matrix is not guaranteed to be positive definite,[2] sampling-based low-rank approximation is a natural approach for scalability, as previously noted by de Silva and Tenenbaum (2003) in the context of the Nyström method. Hence, in this work we evaluate the efficacy of the Nyström method and the Column sampling method on the task of large-scale manifold learning. We also discuss our efficient implementation of a scalable manifold learning pipeline that leverages modern distributed computing architecture in order to construct neighborhood graphs, calculate shortest paths within these graphs and finally compute large-scale low-rank matrix approximations.

We now summarize our main contributions. First, we show connections between two random sampling based singular value decomposition algorithms and provide the first direct comparison of their performances on a variety of approximation tasks. In particular, we show that the Column sampling method is superior for approximating singular values, singular vectors and matrix projection approximations (defined in Section 3.2), while the Nyström method is better for spectral reconstruction approximations (also defined in Section 3.2), which are most relevant in the context of low-rank approximation of large dense matrices. Second, we apply these two algorithms to the task of large-scale manifold learning and present the largest scale study so far on manifold learning, using 18M data points. To date, the largest manifold learning study involves the analysis of music data using 267K points (Platt, 2004).

2. Preliminaries

In this section, we introduce notation and present basic definitions of two of the most common sampling-based techniques for matrix approximation.

2. In the limit of infinite samples, Isomap can be viewed as an instance of Kernel PCA (Ham et al., 2004).

2.1 Notation and Problem Setting

For a matrix $\mathbf{T} \in \mathbb{R}^{a\times b}$, we define $\mathbf{T}^{(j)}$, $j = 1\ldots b$, as the jth column vector of $\mathbf{T}$ and $\mathbf{T}_{(i)}$, $i = 1\ldots a$, as the ith row vector of $\mathbf{T}$. We denote by $\mathbf{T}_k$ the 'best' rank-k approximation to $\mathbf{T}$, that is, $\mathbf{T}_k = \text{argmin}_{\mathbf{V}\in\mathbb{R}^{a\times b},\text{rank}(\mathbf{V})=k}\|\mathbf{T}-\mathbf{V}\|_\xi$, where $\xi \in \{2, F\}$, $\|\cdot\|_2$ denotes the spectral norm and $\|\cdot\|_F$ the Frobenius norm of a matrix. Assuming that $\text{rank}(\mathbf{T}) = r$, we can write the compact Singular Value Decomposition (SVD) of this matrix as $\mathbf{T} = \mathbf{U}_T\mathbf{\Sigma}_T\mathbf{V}_T^\top$ where $\mathbf{\Sigma}_T$ is diagonal and contains the singular values of $\mathbf{T}$ sorted in decreasing order and $\mathbf{U}_T \in \mathbb{R}^{a\times r}$ and $\mathbf{V}_T \in \mathbb{R}^{b\times r}$ are corresponding the left and right singular vectors of $\mathbf{T}$. We can then describe $\mathbf{T}_k$ in terms of its SVD as $\mathbf{T}_k = \mathbf{U}_{T,k}\mathbf{\Sigma}_{T,k}\mathbf{V}_{T,k}^\top$. Let $\mathbf{K} \in \mathbb{R}^{n\times n}$ be a symmetric positive semidefinite (SPSD) kernel or Gram matrix with $\text{rank}(\mathbf{K}) = r \leq n$. We will write the SVD of $\mathbf{K}$ as $\mathbf{K} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^\top$, and the pseudo-inverse of $\mathbf{K}$ as $\mathbf{K}^+ = \sum_{t=1}^r \sigma_t^{-1}\mathbf{U}^{(t)}\mathbf{U}^{(t)\top}$. For $k < r$, $\mathbf{K}_k = \sum_{t=1}^k \sigma_t\mathbf{U}^{(t)}\mathbf{U}^{(t)\top} = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{U}_k^\top$ is the 'best' rank-k approximation to $\mathbf{K}$.

We focus on generating an approximation $\widetilde{\mathbf{K}}$ of $\mathbf{K}$ based on a sample of $l \ll n$ of its columns. We assume that we sample columns uniformly without replacement as suggested by Kumar et al. (2012) and motivated by the connection between uniform sampling and matrix incoherence (Talwalkar and Rostamizadeh, 2010; Mackey et al., 2011), though various methods have been proposed to select columns (see Chapter 4 of Talwalkar (2010) for more details on various sampling schemes). Let $\mathbf{C}$ denote the $n \times l$ matrix formed by these columns and $\mathbf{W}$ the $l \times l$ matrix consisting of the intersection of these l columns with the corresponding l rows of $\mathbf{K}$. Note that $\mathbf{W}$ is SPSD since $\mathbf{K}$ is SPSD. Without loss of generality, the columns and rows of $\mathbf{K}$ can be rearranged based on this sampling so that $\mathbf{K}$ and $\mathbf{C}$ be written as follows:

$$\mathbf{K} = \begin{bmatrix} \mathbf{W} & \mathbf{K}_{21}^\top \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix} \quad \text{and} \quad \mathbf{C} = \begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{21} \end{bmatrix}. \tag{1}$$

The approximation techniques discussed next use the SVD of $\mathbf{W}$ and $\mathbf{C}$ to generate approximations for $\mathbf{K}$.

2.2 Nyström Method

The Nyström method was first introduced as a quadrature method for numerical integration, used to approximate eigenfunction solutions (Nyström, 1928). More recently, it was presented in Williams and Seeger (2000) to speed up kernel algorithms and has been used in applications ranging from manifold learning to image segmentation (Platt, 2004; Fowlkes et al., 2004; Talwalkar et al., 2008). The Nyström method uses $\mathbf{W}$ and $\mathbf{C}$ from (1) to approximate $\mathbf{K}$. Assuming a uniform sampling of the columns, the Nyström method generates a rank-k approximation $\widetilde{\mathbf{K}}$ of $\mathbf{K}$ for $k < n$ defined by:

$$\widetilde{\mathbf{K}}_k^{nys} = \mathbf{C}\mathbf{W}_k^+\mathbf{C}^\top \approx \mathbf{K}, \tag{2}$$

where $\mathbf{W}_k$ is the best k-rank approximation of $\mathbf{W}$ with respect to the spectral or Frobenius norm and $\mathbf{W}_k^+$ denotes the pseudo-inverse of $\mathbf{W}_k$. If we write the SVD of $\mathbf{W}$ as $\mathbf{W} = \mathbf{U}_W\mathbf{\Sigma}_W\mathbf{U}_W^\top$, then from (2) we can write

$$\widetilde{\mathbf{K}}_k^{nys} = \mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}_{W,k}^+\mathbf{U}_{W,k}^\top\mathbf{C}^\top = \left(\sqrt{\frac{l}{n}}\mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}_{W,k}^+\right)\left(\frac{n}{l}\mathbf{\Sigma}_{W,k}\right)\left(\sqrt{\frac{l}{n}}\mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}_{W,k}^+\right)^\top,$$

and hence the Nyström method approximates the top k singular values and vectors of $\mathbf{K}$ as:

$$\widetilde{\mathbf{\Sigma}}_{nys} = \left(\frac{n}{l}\right)\mathbf{\Sigma}_{W,k} \quad \text{and} \quad \widetilde{\mathbf{U}}_{nys} = \sqrt{\frac{l}{n}}\mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}^{+}_{W,k}. \tag{3}$$

The time complexity of compact SVD on $\mathbf{W}$ is in $O(l^2k)$ and matrix multiplication with $\mathbf{C}$ takes $O(nlk)$, hence the total complexity of the Nyström approximation is in $O(nlk)$.

2.3 Column Sampling Method

The Column sampling method was introduced to approximate the SVD of any rectangular matrix (Frieze et al., 1998). It generates approximations of $\mathbf{K}$ by using the SVD of $\mathbf{C}$.[3] If we write the SVD of $\mathbf{C}$ as $\mathbf{C} = \mathbf{U}_C\mathbf{\Sigma}_C\mathbf{V}_C^\top$ then the Column sampling method approximates the top k singular values ($\mathbf{\Sigma}_k$) and singular vectors ($\mathbf{U}_k$) of $\mathbf{K}$ as:

$$\widetilde{\mathbf{\Sigma}}_{col} = \sqrt{\frac{n}{l}}\mathbf{\Sigma}_{C,k} \quad \text{and} \quad \widetilde{\mathbf{U}}_{col} = \mathbf{U}_C = \mathbf{C}\mathbf{V}_{C,k}\mathbf{\Sigma}^{+}_{C,k}. \tag{4}$$

The runtime of the Column sampling method is dominated by the SVD of $\mathbf{C}$. The algorithm takes O(nlk) time to perform compact SVD on $\mathbf{C}$, but is still more expensive than the Nyström method as the constants for SVD are greater than those for the O(nlk) matrix multiplication step in the Nyström method.

3. Nyström Versus Column Sampling

Given that two sampling-based techniques exist to approximate the SVD of SPSD matrices, we pose a natural question: which method should one use to approximate singular values, singular vectors and low-rank approximations? We next analyze the form of these approximations and empirically evaluate their performance in Section 3.3.

3.1 Singular Values and Singular Vectors

As shown in (3) and (4), the singular values of $\mathbf{K}$ are approximated as the scaled singular values of $\mathbf{W}$ and $\mathbf{C}$, respectively. The scaling terms are quite rudimentary and are primarily meant to *compensate* for the 'small sample size' effect for both approximations. Formally, these scaling terms make the approximations in (3) and (4) unbiased estimators of the true singular values. The form of singular vectors is more interesting. The Column sampling singular vectors ($\widetilde{\mathbf{U}}_{col}$) are orthonormal since they are the singular vectors of $\mathbf{C}$. In contrast, the Nyström singular vectors ($\widetilde{\mathbf{U}}_{nys}$) are approximated by *extrapolating* the singular vectors of $\mathbf{W}$ as shown in (3), and are *not* orthonormal. As we show in Section 3.3, this adversely affects the accuracy of singular vector approximation from the Nyström method. It is possible to orthonormalize the Nyström singular vectors by using QR decomposition. Since $\widetilde{\mathbf{U}}_{nys} \propto \mathbf{C}\mathbf{U}_W\mathbf{\Sigma}^{+}_W$, where $\mathbf{U}_W$ is orthogonal and $\mathbf{\Sigma}_W$ is diagonal, this simply implies that QR decomposition creates an orthonormal span of $\mathbf{C}$ rotated by $\mathbf{U}_W$. However, the complexity of QR decomposition of $\widetilde{\mathbf{U}}_{nys}$ is the same as that of the SVD of $\mathbf{C}$. Thus, the computational cost of orthogonalizing $\widetilde{\mathbf{U}}_{nys}$ would nullify the computational benefit of the Nyström method over Column sampling.

3. The Nyström method also uses sampled columns of $\mathbf{K}$, but the Column sampling method is named so because it uses direct decomposition of $\mathbf{C}$, while the Nyström method decomposes its submatrix, $\mathbf{W}$.

3.2 Low-rank Approximation

Several studies have empirically shown that the accuracy of low-rank approximations of kernel matrices is tied to the performance of kernel-based learning algorithms (Williams and Seeger, 2000; Talwalkar and Rostamizadeh, 2010). Furthermore, the connection between kernel matrix approximation and the *hypothesis* generated by several widely used kernel-based learning algorithms has been theoretically analyzed (Cortes et al., 2010). Hence, accurate low-rank approximations are of great practical interest in machine learning. As discussed in Section 2.1, the optimal $\mathbf{K}_k$ is given by,

$$\mathbf{K}_k = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{U}_k^\top = \mathbf{U}_k \mathbf{U}_k^\top \mathbf{K} = \mathbf{K} \mathbf{U}_k \mathbf{U}_k^\top$$

where the columns of $\mathbf{U}_k$ are the k singular vectors of $\mathbf{K}$ corresponding to the top k singular values of $\mathbf{K}$. We refer to $\mathbf{U}_k \mathbf{\Sigma}_k \mathbf{U}_k^\top$ as *Spectral Reconstruction*, since it uses both the singular values and vectors of $\mathbf{K}$, and $\mathbf{U}_k \mathbf{U}_k^\top \mathbf{K}$ as *Matrix Projection*, since it uses only singular vectors to compute the projection of $\mathbf{K}$ onto the space spanned by vectors $\mathbf{U}_k$. These two low-rank approximations are equal only if $\mathbf{\Sigma}_k$ and $\mathbf{U}_k$ contain the true singular values and singular vectors of $\mathbf{K}$. Since this is not the case for approximate methods such as Nyström and Column sampling these two measures generally give different errors. Thus, we analyze each measure separately in the following sections.

3.2.1 MATRIX PROJECTION

For Column sampling using (4), the low-rank approximation via matrix projection is

$$\widetilde{\mathbf{K}}_k^{col} = \widetilde{\mathbf{U}}_{col,k} \widetilde{\mathbf{U}}_{col,k}^\top \mathbf{K} = \mathbf{U}_{C,k} \mathbf{U}_{C,k}^\top \mathbf{K} = \mathbf{C}((\mathbf{C}^\top \mathbf{C})_k)^+ \mathbf{C}^\top \mathbf{K}, \tag{5}$$

where $(\mathbf{C}^\top \mathbf{C})_k^{-1} = \mathbf{V}_{C,k} (\mathbf{\Sigma}_{C,k}^2)^+ \mathbf{V}_{C,k}^\top$. Clearly, if $k = l$, $(\mathbf{C}^\top \mathbf{C})_k = \mathbf{C}^\top \mathbf{C}$. Similarly, using (3), the Nyström matrix projection is

$$\widetilde{\mathbf{K}}_k^{nys} = \widetilde{\mathbf{U}}_{nys,k} \widetilde{\mathbf{U}}_{nys,k}^\top \mathbf{K} = \frac{l}{n} \mathbf{C} (\mathbf{W}_k^2)^+ \mathbf{C}^\top \mathbf{K}. \tag{6}$$

As shown in (5) and (6), the two methods have similar expressions for matrix projection, except that $\mathbf{C}^\top \mathbf{C}$ is replaced by a scaled $\mathbf{W}^2$. The scaling term appears only in the expression for the Nyström method. We now present Theorem 1 and Observations 1 and 2, which provide further insights about these two methods in the context of matrix projection.

Theorem 1 *The Column sampling and Nyström matrix projections are of the form* $\mathbf{U}_C \mathbf{R} \mathbf{U}_C^\top \mathbf{K}$*, where* $\mathbf{R} \in \mathbb{R}^{l \times l}$ *is SPSD. Further, Column sampling gives the lowest reconstruction error* (*measured in* $\|\cdot\|_F$) *among all such approximations if* $k = l$.

Observation 1 *For* $k = l$*, matrix projection for Column sampling reconstructs* $\mathbf{C}$ *exactly. This can be seen by block-decomposing* $\mathbf{K}$ *as:* $\mathbf{K} = [\mathbf{C} \quad \bar{\mathbf{C}}]$*, where* $\bar{\mathbf{C}} = [\mathbf{K}_{21} \quad \mathbf{K}_{22}]^\top$*, and using* (5)*:*

$$\widetilde{\mathbf{K}}_l^{col} = \mathbf{C}(\mathbf{C}^\top \mathbf{C})^+ \mathbf{C}^\top \mathbf{K} = [\mathbf{C} \quad \mathbf{C}(\mathbf{C}^\top \mathbf{C})^+ \mathbf{C}^\top \bar{\mathbf{C}}] = [\mathbf{C} \quad \bar{\mathbf{C}}].$$

Observation 2 *For* $k = l$*, the span of the orthogonalized Nyström singular vectors equals the span of* $\widetilde{\mathbf{U}}_{col}$*, as discussed in Section 3.1. Hence, matrix projection is identical for Column sampling and Orthonormal Nyström for* $k = l$.

Matrix projection approximations are not necessarily symmetric and require storage of and multiplication with $\mathbf{K}$. Hence, although matrix projection is often analyzed theoretically, for large-scale problems, the storage and computational requirements may be inefficient or even infeasible.

Data Set	Data	n	d	Kernel
PIE-2.7K	faces	2731	2304	linear
PIE-7K	faces	7412	2304	linear
MNIST	digits	4000	784	linear
ESS	proteins	4728	16	RBF
ABN	abalones	4177	8	RBF

Table 1: Description of the data sets used in our experiments comparing sampling-based matrix approximations (Sim et al., 2002; LeCun and Cortes, 1998; Talwalkar et al., 2008). 'n' denotes the number of points and 'd' denotes the data dimensionality, that is, the number of features in input space.

3.2.2 SPECTRAL RECONSTRUCTION

Using (3), the Nyström spectral reconstruction is:

$$\widetilde{\mathbf{K}}_k^{nys} = \widetilde{\mathbf{U}}_{nys,k}\widetilde{\mathbf{\Sigma}}_{nys,k}\widetilde{\mathbf{U}}_{nys,k}^\top = \mathbf{C}\mathbf{W}_k^+\mathbf{C}^\top. \tag{7}$$

When $k=l$, this approximation perfectly reconstructs three blocks of $\mathbf{K}$, and $\mathbf{K}_{22}$ is approximated by the Schur Complement of $\mathbf{W}$ in $\mathbf{K}$. The Column sampling spectral reconstruction has a similar form as (7):

$$\widetilde{\mathbf{K}}_k^{col} = \widetilde{\mathbf{U}}_{col,k}\widetilde{\mathbf{\Sigma}}_{col,k}\widetilde{\mathbf{U}}_{col,k}^\top = \sqrt{n/l}\,\mathbf{C}\big((\mathbf{C}^\top\mathbf{C})_k^{\frac{1}{2}}\big)^+\mathbf{C}^\top. \tag{8}$$

In contrast with matrix projection, the scaling term now appears in the Column sampling reconstruction. To analyze the two approximations, we consider an alternative characterization using the fact that $\mathbf{K} = \mathbf{X}^\top\mathbf{X}$ for some $\mathbf{X} \in \mathbb{R}^{N\times n}$. We define a zero-one sampling matrix, $\mathbf{S} \in \mathbb{R}^{n\times l}$, that selects l columns from $\mathbf{K}$, that is, $\mathbf{C} = \mathbf{K}\mathbf{S}$. Further, $\mathbf{W} = \mathbf{S}^\top\mathbf{K}\mathbf{S} = \mathbf{X}'^\top\mathbf{X}'$, where $\mathbf{X}' \in \mathbb{R}^{N\times l}$ contains l sampled columns of $\mathbf{X}$ and $\mathbf{X}' = \mathbf{U}_{X'}\mathbf{\Sigma}_{X'}\mathbf{V}_{X'}^\top$ is the SVD of $\mathbf{X}'$. We now present two results. Theorem 2 shows that the optimal spectral reconstruction is data dependent and may differ from the Nyström and Column sampling approximations. Moreover, Theorem 3 reveals that in certain instances the Nyström method is optimal, while the Column sampling method enjoys no such guarantee.

Theorem 2 *Column sampling and Nyström spectral reconstructions of rank k are of the form $\mathbf{X}^\top\mathbf{U}_{X',k}\mathbf{Z}\mathbf{U}_{X',k}^\top\mathbf{X}$, where $\mathbf{Z} \in \mathbb{R}^{k\times k}$ is SPSD. Further, among all approximations of this form, neither the Column sampling nor the Nyström approximation is optimal (in $\|\cdot\|_F$).*

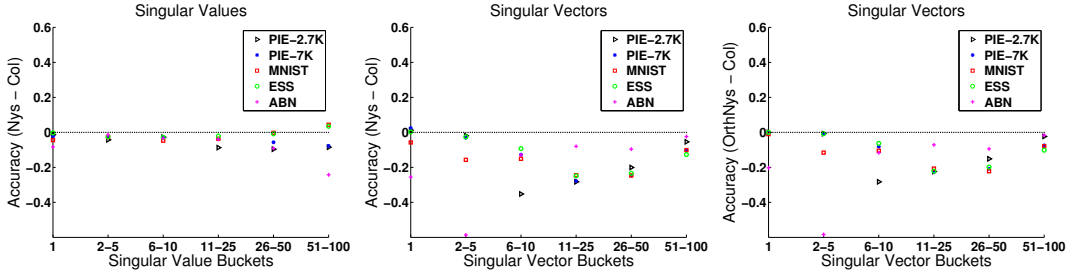
Theorem 3 *Let $r = \mathrm{rank}(\mathbf{K}) \le k \le l$ and $\mathrm{rank}(\mathbf{W}) = r$. Then, the Nyström approximation is exact for spectral reconstruction. In contrast, Column sampling is exact iff $\mathbf{W} = \big((l/n)\mathbf{C}^\top\mathbf{C}\big)^{1/2}$.*

3.3 Empirical Comparison

To test the accuracy of singular values/vectors and low-rank approximations for different methods, we used several kernel matrices arising in different applications, as described in Table 3.3. We worked with data sets containing less than ten thousand points to be able to compare with exact SVD. We fixed k to be 100 in all the experiments, which captures more than 90% of the spectral energy for each data set.

For singular values, we measured percentage accuracy of the approximate singular values with respect to the exact ones. For a fixed l, we performed 10 trials by selecting columns uniformly at random from $\mathbf{K}$. We show in Figure 1(a) the difference in mean percentage accuracy for the two methods for $l = n/10$, with results bucketed by groups of singular values, that is, we sorted the singular values in descending order, grouped them as indicated in the figure, and report the average percentage accuracy for each group. The empirical results show that the Column sampling method generates more accurate singular values than the Nyström method. A similar trend was observed for other values of l.

For singular vectors, the accuracy was measured by the dot product, that is, cosine of principal angles between the exact and the approximate singular vectors. Figure 1(b) shows the difference in mean accuracy between Nyström and Column sampling methods, once again bucketed by groups of singular vectors sorted in descending order based on their corresponding singular values. The top 100 singular vectors were all better approximated by Column sampling for all data sets. This trend was observed for other values of l as well. Furthermore, even when the Nyström singular vectors are orthogonalized, the Column sampling approximations are superior, as shown in Figure 1(c).



(a) (b) (c)

Figure 1: Comparison of singular values and vectors (values above zero indicate better performance of Nyström). (a) Top 100 singular values with $l = n/10$. (b) Top 100 singular vectors with $l = n/10$. (c) Comparison using orthogonalized Nyström singular vectors.

Next we compared the low-rank approximations generated by the two methods using matrix projection and spectral reconstruction. We measured the accuracy of reconstruction relative to the optimal rank-k approximation, $\mathbf{K}_k$, via relative accuracy $= \frac{\|\mathbf{K}-\mathbf{K}_k\|_F}{\|\mathbf{K}-\widetilde{\mathbf{K}}_k^{nys/col}\|_F}$, which will approach one for good approximations. As motivated by Theorem 1, Column sampling generates better reconstructions via matrix projection (Figure 2(a)). In contrast, the Nyström method produces superior results for spectral reconstruction (Figure 2(b)). These results may appear somewhat surprising given the relatively poor quality of the singular values/vectors for the Nyström method, but they are in agreement with the consequences of Theorem 3 and the fact that the kernel matrices we consider (aside from 'DEXT') are nearly low-rank. Moreover, the performance of these two approximations are indeed tied to the spectrum of $\mathbf{K}$ as stated in Theorem 2. Indeed, we found that the two approximations were roughly equivalent for a sparse kernel matrix with slowly decaying spectrum ('DEXT' in Figure 2(b)), while the Nyström method was superior for dense kernel matrices with exponentially decaying spectra arising from the other data sets used in the experiments.

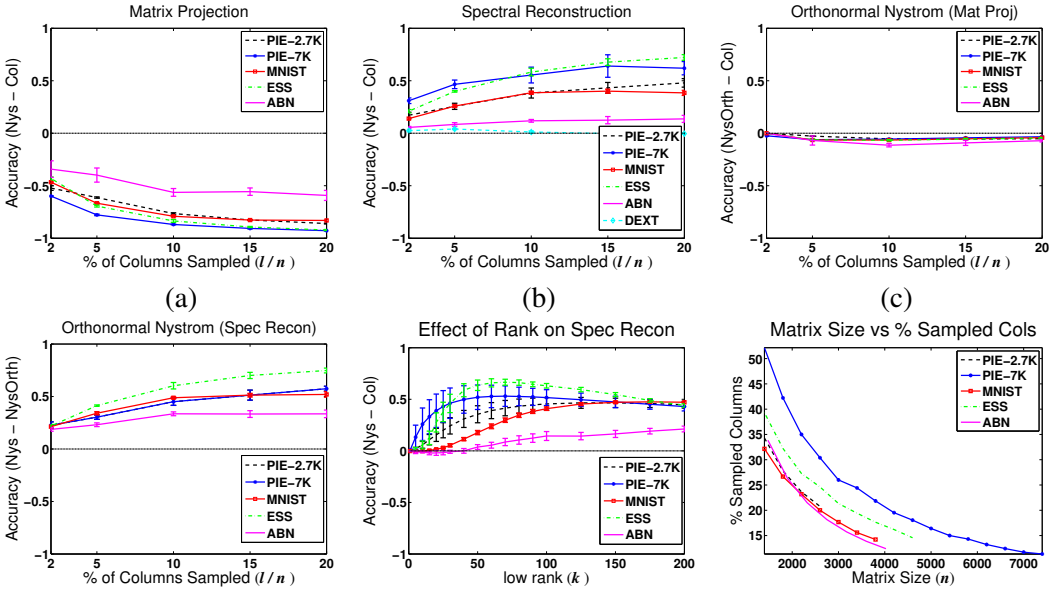


Figure 2: Plots (a)-(e) compare performance of various matrix approximations, and in all cases values above zero indicate better performance of the first listed method. (a) Matrix projection, Nyström versus Column sampling, $k = 100$. (b) Spectral reconstruction, Nyström versus Column sampling, $k = 100$. (c) Matrix projection, Orthonormal Nyström versus Column sampling, $k = 100$. (d) Spectral reconstruction, Nyström versus Orthonormal Nyström, $k = 100$. (e) Spectral reconstruction, Nyström versus Column sampling varying ranks. (f) Percentage of columns (l/n) needed to achieve 75% relative accuracy for Nyström spectral reconstruction as a function of n.

The non-orthonormality of the Nyström method's singular vectors (Section 3.1) is one factor that impacts its accuracy for some tasks. When orthonormalized, the Nyström matrix projection error is reduced considerably as shown in Figure 2(c), and supported by Observation 2. Also, the accuracy of Orthonormal Nyström spectral reconstruction is worse relative to the standard Nyström approximation, as shown in Figure 2(d). This result can be attributed to the fact that orthonormalization of the singular vectors leads to the loss of some of the unique properties described in Section 3.2.2. For instance, Theorem 3 no longer holds and the scaling terms do not cancel out, that is, $\widetilde{\mathbf{K}}_k^{nys} \neq \mathbf{C}\mathbf{W}_k^{+}\mathbf{C}^{\top}$.

We next tested the accuracy of spectral reconstruction for the two methods for varying values of k and a fixed l. We found that the Nyström method outperforms Column sampling across all tested values of k, as shown in Figure 2(e). Next, we addressed another basic issue: how many columns do we need to obtain reasonable reconstruction accuracy? We performed an experiment in which we fixed k and varied the size of our data set (n). For each n, we performed grid search over l to find the minimal l for which the relative accuracy of Nyström spectral reconstruction was at least 75%.

Figure 2(f) shows that the required percentage of columns (l/n) decreases quickly as n increases, lending support to the use of sampling-based algorithms for large-scale data.

Finally, we note another important distinction between the Nyström method and Column sampling, namely, out-of-sample extension. Out-of-sample extension is often discussed in the context of manifold learning, in which case it involves efficiently deriving a low-dimensional embedding for an arbitrary test point given embeddings from a set of training points (rather than rerunning the entire manifold learning algorithm). The Nyström method naturally lends itself to such out-of-sample extension, as a new point can be processed based on extrapolating from the sampled points (de Silva and Tenenbaum, 2003). Moreover, it is possible to use Column sampling to learn embeddings on the initial sample, and then use the Nyström method for subsequent out-of-sample-extension. Hence, given a large set of samples, both the Nyström method and Column sampling are viable options to enhance the scalability of manifold learning methods, as we will explore in Section 4.

4. Large-scale Manifold Learning

In the previous section, we discussed two sampling-based techniques that generate approximations for kernel matrices. Although we analyzed the effectiveness of these techniques for approximating singular values, singular vectors and low-rank matrix reconstruction, we have yet to discuss the effectiveness of these techniques in the context of actual machine learning tasks. In fact, the Nyström method has been shown to be successful on a variety of learning tasks including Support Vector Machines (Fine and Scheinberg, 2002), Gaussian Processes (Williams and Seeger, 2000), Spectral Clustering (Fowlkes et al., 2004), Kernel Ridge Regression (Cortes et al., 2010) and more generally to approximate regularized matrix inverses via the Woodbury approximation (Williams and Seeger, 2000). In this section, we will discuss how approximate embeddings can be used in the context of manifold learning, relying on the sampling based algorithms from the previous section to generate an approximate SVD. We present the largest study to date for manifold learning, and provide a quantitative comparison of Isomap and Laplacian Eigenmaps for large scale face manifold construction on clustering and classification tasks.

4.1 Manifold Learning

Manifold learning aims to extract low-dimensional structure from high-dimensional data. Given n input points, $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ and $\mathbf{x}_i \in \mathbb{R}^d$, the goal is to find corresponding outputs $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n$, where $\mathbf{y}_i \in \mathbb{R}^k$, $k \ll d$, such that $\mathbf{Y}$ 'faithfully' represents $\mathbf{X}$. Several manifold learning techniques have been proposed, for example, Isomap (Tenenbaum et al., 2000), Laplacian Eigenmaps (Belkin and Niyogi, 2001), Local Linear Embedding (LLE) (Roweis and Saul, 2000), Hessian Eigenmaps (Donoho and Grimes, 2003), Structural Preserving Embedding (SPE) (Shaw and Jebara, 2009) and Semidefinite Embedding (SDE) (Weinberger and Saul, 2006). Isomap aims to preserve all pair-wise geodesic distances, whereas LLE, Laplacian Eigenmaps and Hessian Eigenmaps focus on preserving local neighborhood relationships. SDE and SPE are both formulated as instances of semidefinite programming, and are prohibitively expensive for large-scale problems. We will focus on the Isomap and Laplacian Eigenmaps algorithms as they are well-studied and highlight the differences between global versus local manifold learning techniques. We next briefly review the main computational efforts required for both algorithms, which involve neighborhood graph construction and manipulation and SVD of a symmetric similarity matrix.

4.1.1 ISOMAP

Isomap aims to extract a low-dimensional data representation that best preserves all pairwise distances between input points, as measured by their geodesic distances along the manifold (Tenenbaum et al., 2000). It approximates the geodesic distance assuming that input space distance provides good approximations for nearby points, and for faraway points it estimates distance as a series of hops between neighboring points. Isomap involves three steps: i) identifying t nearest neighbors for each point and constructing the associated undirected neighborhood graph in $O(n^2)$ time, ii) computing approximate geodesic distances via the neighborhood graph and converting these distances into a similarity matrix $\mathbf{K}$ via double centering, which overall requires $O(n^2 \log n)$ time, and iii) calculating the final embedding of the form $\mathbf{Y} = (\mathbf{\Sigma}_k)^{1/2}\mathbf{U}_k^\top$, where $\mathbf{\Sigma}_k$ is the diagonal matrix of the top k singular values of $\mathbf{K}$ and $\mathbf{U}_k$ are the associated singular vectors, which requires $O(n^2)$ space for storing $\mathbf{K}$, and $O(n^3)$ time for its SVD. The time and space complexities for all three steps are intractable for $n = 18$M.

4.1.2 LAPLACIAN EIGENMAPS

Laplacian Eigenmaps aims to find a low-dimensional representation that best preserves local neighborhood relations (Belkin and Niyogi, 2001). The algorithm first computes t nearest neighbors for each point, from which it constructs a sparse weight matrix $\mathbf{W}$.[4] It then minimizes an objective function that penalizes nearby inputs for being mapped to faraway outputs, with 'nearness' measured by the weight matrix $\mathbf{W}$, and the solution to the objective is the bottom singular vectors of the symmetrized, normalized form of the graph Laplacian, $\mathcal{L}$. The runtime of this algorithm is dominated by computing nearest neighbors, since the subsequent steps involve sparse matrices. In particular, $\mathcal{L}$ can be stored in $O(tn)$ space, and iterative methods, such as Lanczos, can be used to compute its compact SVD relatively quickly.

4.2 Approximation Experiments

Since we use sampling-based SVD approximation to scale Isomap, we first examined how well the Nyström and Column sampling methods approximated our desired low-dimensional embeddings, that is, $\mathbf{Y} = (\mathbf{\Sigma}_k)^{1/2}\mathbf{U}_k^\top$. Using (3), the Nyström low-dimensional embeddings are:

$$\widetilde{\mathbf{Y}}^{nys} = \widetilde{\mathbf{\Sigma}}_{nys,k}^{1/2}\widetilde{\mathbf{U}}_{nys,k}^\top = \left((\mathbf{\Sigma}_W)_k^{1/2}\right)^+ \mathbf{U}_{W,k}^\top \mathbf{C}^\top.$$

Similarly, from (4) we can express the Column sampling low-dimensional embeddings as:

$$\widetilde{\mathbf{Y}}^{col} = \widetilde{\mathbf{\Sigma}}_{col,k}^{1/2}\widetilde{\mathbf{U}}_{col,k}^\top = \sqrt[4]{\frac{n}{l}}\left((\mathbf{\Sigma}_C)_k^{1/2}\right)^+ \mathbf{V}_{C,k}^\top \mathbf{C}^\top.$$

Both approximations are of a similar form. Further, notice that the optimal low-dimensional embeddings are in fact the square root of the optimal rank k approximation to the associated SPSD matrix, that is, $\mathbf{Y}^\top\mathbf{Y} = \mathbf{K}_k$, for Isomap. As such, there is a connection between the task of approximating low-dimensional embeddings and the task of generating low-rank approximate spectral reconstructions, as discussed in Section 3.2.2. Recall that the theoretical analysis in Section 3.2.2

4. The weight matrix should not be confused with the subsampled SPSD matrix, $\mathbf{W}$, associated with the Nyström method. Since sampling-based approximation techniques will not be used with Laplacian Eigenmaps, the notation should be clear from the context.

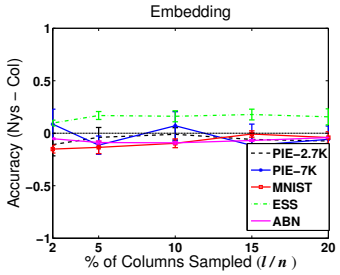


Figure 3: Comparison of embeddings (values above zero indicate better performance of Nyström).

as well as the empirical results in Section 3.3 both suggested that the Nyström method was superior in its spectral reconstruction accuracy. Hence, we performed an empirical study using the data sets from Table 3.3 to measure the quality of the low-dimensional embeddings generated by the two techniques and see if the same trend exists.

We measured the quality of the low-dimensional embeddings by calculating the extent to which they preserve distances, which is the appropriate criterion in the context of manifold learning. For each data set, we started with a kernel matrix, $\mathbf{K}$, from which we computed the associated $n \times n$ squared distance matrix, $\mathbf{D}$, using the fact that $\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \mathbf{K}_{ii} + \mathbf{K}_{jj} - 2\mathbf{K}_{ij}$. We then computed the approximate low-dimensional embeddings using the Nyström and Column sampling methods, and then used these embeddings to compute the associated approximate squared distance matrix, $\widetilde{\mathbf{D}}$. We measured accuracy using the notion of relative accuracy defined in Section 3.3.

In our experiments, we set $k = 100$ and used various numbers of sampled columns, ranging from $l = n/50$ to $l = n/5$. Figure 3 presents the results of our experiments. Surprisingly, we do not see the same trend in our empirical results for embeddings as we previously observed for spectral reconstruction, as the two techniques exhibit roughly similar behavior across data sets. As a result, we decided to use both the Nyström and Column sampling methods for our subsequent manifold learning study.

4.3 Large-scale Learning

In this section, we outline the process of learning a manifold of faces. We first describe the data sets used in our experiments. We then explain how to extract nearest neighbors, a common step between Laplacian Eigenmaps and Isomap. The remaining steps of Laplacian Eigenmaps are straightforward, so the subsequent sections focus on Isomap, and specifically on the computational efforts required to generate a manifold using Webfaces-18M.[5]

4.3.1 DATA SETS

We used two faces data sets consisting of 35K and 18M images. The CMU PIE face data set (Sim et al., 2002) contains 41,368 images of 68 subjects under 13 different poses and various illumination conditions. A standard face detector extracted 35,247 faces (each 48×48 pixels), which comprised our 35K set (PIE-35K). Being labeled, this set allowed us to perform quantitative comparisons. The

5. To run Laplacian Eigenmaps, we generated $\mathbf{W}$ from nearest neighbor data for the largest component of the neighborhood graph and used a sparse eigensolver to compute the bottom eigenvalues of $\mathcal{L}$.

second data set, named Webfaces-18M, contains 18.2 million images extracted from the Web using the same face detector. For both data sets, face images were represented as 2304 dimensional pixel vectors that were globally normalized to have zero mean and unit variance. No other pre-processing, for example, face alignment, was performed. In contrast, He et al. (2005) used well-aligned faces (as well as much smaller data sets) to learn face manifolds. Constructing Webfaces-18M, including face detection and duplicate removal, took 15 hours using a cluster of 500 machines. We used this cluster for all experiments requiring distributed processing and data storage.

4.3.2 NEAREST NEIGHBORS AND NEIGHBORHOOD GRAPH

The cost of naive nearest neighbor computation is $O(n^2)$, where n is the size of the data set. It is possible to compute exact neighbors for PIE-35K, but for Webfaces-18M this computation is prohibitively expensive. So, for this set, we used a combination of random projections and spill trees (Liu et al., 2004) to get approximate neighbors. Computing 5 nearest neighbors in parallel with spill trees took $\sim$2 days on the cluster. Figure 4(a) shows the top 5 neighbors for a few randomly chosen images in Webfaces-18M. In addition to this visualization, comparison of exact neighbors and spill tree approximations for smaller subsets suggested good performance of spill trees.

We next constructed the neighborhood graph by representing each image as a node and connecting all neighboring nodes. Since Isomap and Laplacian Eigenmaps require this graph to be connected, we used depth-first search to find its largest connected component. These steps required $O(tn)$ space and time. Constructing the neighborhood graph for Webfaces-18M and finding the largest connected component took 10 minutes on a single machine using the OpenFST library (Allauzen et al., 2007).

For neighborhood graph construction, an 'appropriate' choice of number of neighbors, t, is crucial. A small t may give too many disconnected components, while a large t may introduce unwanted edges. These edges stem from inadequately sampled regions of the manifold and false positives introduced by the face detector. Since Isomap needs to compute shortest paths in the neighborhood graph, the presence of bad edges can adversely impact these computations. This is known as the problem of leakage or 'short-circuits' (Balasubramanian and Schwartz, 2002). Here, we chose $t = 5$ and also enforced an upper limit on neighbor distance to alleviate the problem of leakage. We used a distance limit corresponding to the 95^{th} percentile of neighbor distances in the PIE-35K data set. Figure 4(b) shows the effect of choosing different values for t with and without enforcing the upper distance limit. As expected, the size of the largest connected component increases as t increases. Also, enforcing the distance limit reduces the size of the largest component. See Appendix D for visualizations of various components of the neighborhood graph.

4.3.3 APPROXIMATING GEODESICS

To construct the similarity matrix $\mathbf{K}$ in Isomap, one approximates geodesic distance by shortest-path lengths between every pair of nodes in the neighborhood graph. This requires $O(n^2 \log n)$ time and $O(n^2)$ space, both of which are prohibitive for 18M nodes. However, since we use sampling-based approximate decomposition, we need only $l \ll n$ columns of $\mathbf{K}$, which form the submatrix $\mathbf{C}$. We thus computed geodesic distance between l randomly selected nodes (called landmark points) and the rest of the nodes, which required $O(ln \log n)$ time and $O(ln)$ space. Since this computation can easily be parallelized, we performed geodesic computation on the cluster and stored the output in a distributed fashion. The overall procedure took 60 minutes for Webfaces-18M using $l = 10$K. The

(a)

	No Upper Limit		Upper Limit Enforced	
t	# Comp	% Largest	# Comp	% Largest
1	1.7M	0.05 %	4.3M	0.03 %
2	97K	97.2 %	285K	80.1 %
3	18K	99.3 %	277K	82.2 %
5	1.9K	99.9 %	275K	83.1 %

(b)

Figure 4: (a) Visualization of neighbors for Webfaces-18M. The first image in each row is the input, and the next five are its neighbors. (b) Number of components in the Webfaces-18M neighbor graph and the percentage of images within the largest connected component ('% Largest') for varying numbers of neighbors (t) with and without an upper limit on neighbor distances.

bottom four rows in Figure 5 show sample shortest paths for images within the largest component for Webfaces-18M, illustrating smooth transitions between images along each path.[6]

4.3.4 GENERATING LOW-DIMENSIONAL EMBEDDINGS

Before generating low-dimensional embeddings using Isomap, one needs to convert distances into similarities using a process called centering (Tenenbaum et al., 2000). For the Nyström approximation, we computed $\mathbf{W}$ by double centering $\mathbf{D}$, the $l \times l$ matrix of squared geodesic distances between all landmark nodes, as $\mathbf{W} = -\frac{1}{2}\mathbf{HDH}$, where $\mathbf{H} = \mathbf{I}_l - \frac{1}{l}\mathbf{1}\mathbf{1}^\top$ is the centering matrix, $\mathbf{I}_l$ is the $l \times l$ identity matrix and $\mathbf{1}$ is a column vector of all ones. Similarly, the matrix $\mathbf{C}$ was obtained from squared geodesic distances between the landmark nodes and all other nodes using single-centering as described in de Silva and Tenenbaum (2003).

For the Column sampling approximation, we decomposed $\mathbf{C}^\top\mathbf{C}$, which we constructed by performing matrix multiplication in parallel on $\mathbf{C}$. For both approximations, decomposition on an $l \times l$ matrix ($\mathbf{C}^\top\mathbf{C}$ or $\mathbf{W}$) took about one hour. Finally, we computed low-dimensional embeddings by multiplying the scaled singular vectors from approximate decomposition with $\mathbf{C}$. For Webfaces-18M, generating low dimensional embeddings took 1.5 hours for the Nyström method and 6 hours for the Column sampling method.

4.4 Manifold Evaluation

Manifold learning techniques typically transform the data such that Euclidean distance in the transformed space between *any* pair of points is meaningful, under the assumption that in the original space Euclidean distance is meaningful only in local neighborhoods. Since K-means clustering computes Euclidean distances between all pairs of points, it is a natural choice for evaluating these techniques. We also compared the performance of various techniques using nearest neighbor classification. Since CMU-PIE is a labeled data set, we first focused on quantitative evaluation of different

6. Our techniques for approximating geodesic distances via shortest path are used by Google for its "People Hopper" application which runs on the social networking site Orkut (Kumar and Rowley, 2010).

Methods	Purity (%)	Accuracy (%)
PCA	54.3 (±0.8)	46.1 (±1.4)
Isomap	58.4 (±1.1)	53.3 (±4.3)
Nys-Iso	59.1 (±0.9)	53.3 (±2.7)
Col-Iso	56.5 (±0.7)	49.4 (±3.8)
Lap. Eig.	35.8 (±5.0)	69.2 (±10.8)

(a)

Methods	Purity (%)	Accuracy (%)
PCA	54.6 (±1.3)	46.8 (±1.3)
Nys-Iso	59.9 (±1.5)	53.7 (±4.4)
Col-Iso	56.1 (±1.0)	50.7 (±3.3)
Lap. Eig.	39.3 (±4.9)	74.7 (±5.1)

(b)

Table 2: K-means clustering of face poses. Results are averaged over 10 random K-means initializations. (a) PIE-10K. (b) PIE-35K.

embeddings using face pose as class labels. The PIE set contains faces in 13 poses, and such a fine sampling of the pose space makes clustering and classification tasks very challenging. In all the experiments we fixed the dimension of the reduced space, k, to be 100.

We first compared different Isomap approximations to exact Isomap, using a subset of PIE with 10K images (PIE-10K) so that exact SVD required by Isomap was feasible. We fixed the number of clusters in our experiments to equal the number of pose classes, and measured clustering performance using two measures, *Purity* and *Accuracy*. Purity measures the frequency of data belonging to the same cluster sharing the same class label, while Accuracy measures the frequency of data from the same class appearing in a single cluster. Table 2(a) shows that clustering with Nyström Isomap with just $l = 1$K performs almost as well as exact Isomap on this data set. Column sampling Isomap performs slightly worse than Nyström Isomap. The clustering results on the full PIE-35K set (Table 2(b)) with $l = 10$K affirm this observation. As illustrated by Figure 7 (Appendix E), the Nyström method separates the pose clusters better than Column sampling verifying the quantitative results in Table 2.

One possible reason for the poor performance of Column sampling Isomap is due to the form of the similarity matrix $\mathbf{K}$. When using a finite number of data points for Isomap, $\mathbf{K}$ is not guaranteed to be SPSD (Ham et al., 2004). We verified that $\mathbf{K}$ was not SPSD in our experiments, and a significant number of top eigenvalues, that is, those with largest magnitudes, were negative. The two approximation techniques differ in their treatment of negative eigenvalues and the corresponding eigenvectors. The Nyström method allows one to use eigenvalue decomposition (EVD) of $\mathbf{W}$ to yield signed eigenvalues, making it possible to discard the negative eigenvalues and the corresponding eigenvectors. It is not possible to discard these in the Column-based method, since the signs of eigenvalues are lost in the SVD of the rectangular matrix $\mathbf{C}$ (or EVD of $\mathbf{C}^\top\mathbf{C}$). Thus, the presence of negative eigenvalues deteriorates the performance of Column sampling method more than the Nyström method.

Table 2(a) and 2(b) also show a significant difference in the Isomap and Laplacian Eigenmaps results. The 2D embeddings of PIE-35K (Figure 7 in Appendix E) reveal that Laplacian Eigenmaps projects data points into a small compact region. When used for clustering, these compact embeddings lead to a few large clusters and several tiny clusters, thus explaining the high accuracy and low purity of the clusters. This indicates poor clustering performance of Laplacian Eigenmaps, since one can achieve even 100% Accuracy simply by grouping all points into a single cluster. However, the Purity of such clustering would be very low. Finally, the improved clustering results of Isomap over PCA for both data sets verify that the manifold of faces is not linear in the input space.

Methods	$K=1$	$K=3$	$K=5$
Isomap	10.9 ($\pm$0.5)	14.1 ($\pm$0.7)	15.8 ($\pm$0.3)
Nys-Iso	11.0 ($\pm$0.5)	14.0 ($\pm$0.6)	15.8 ($\pm$0.6)
Col-Iso	12.0 ($\pm$0.4)	15.3 ($\pm$0.6)	16.6 ($\pm$0.5)
Lap. Eig.	12.7 ($\pm$0.7)	16.6 ($\pm$0.5)	18.9 ($\pm$0.9)

(a)

Nys-Iso	Col-Iso	Lap. Eig.
9.8 ($\pm$0.2)	10.3 ($\pm$0.3)	11.1 ($\pm$0.3)

(b)

Table 3: Nearest neighbor face pose classification error (%). Results are averaged over 10 random splits of training and test sets. (a) PIE-10K with $K = \{1,3,5\}$ neighbors. (b) PIE-35K with $K = 1$ neighbors.

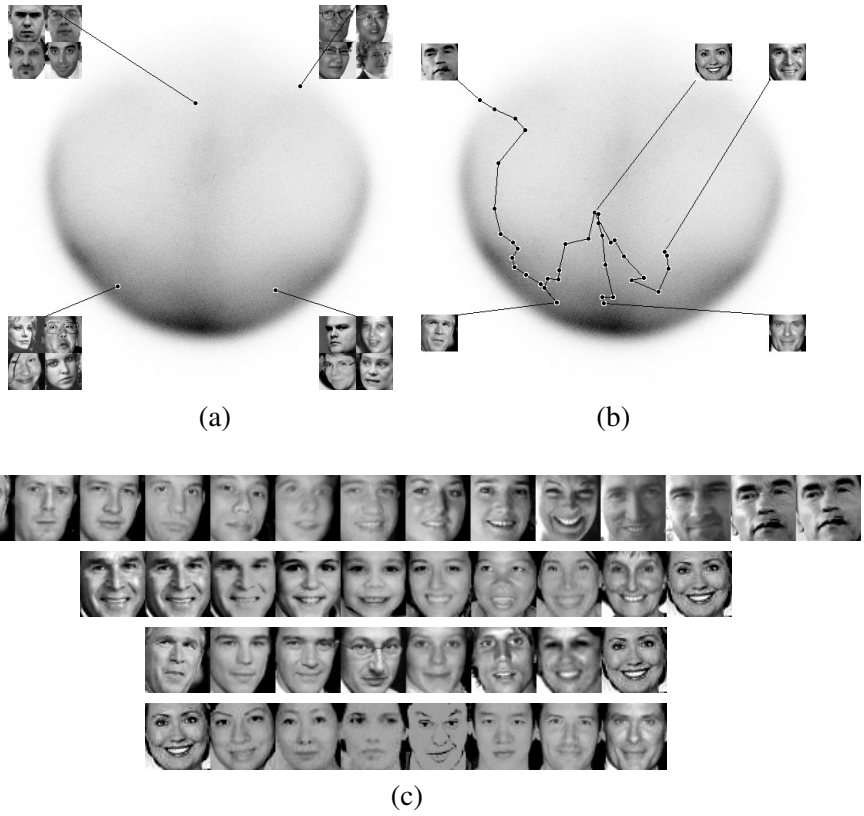


Figure 5: 2D embedding of Webfaces-18M using Nyström Isomap (Top row). Darker areas indicate denser manifold regions. (a) Face samples at different locations on the manifold. (b) Approximate geodesic paths between celebrities. (c) Visualization of paths shown in (b).

Moreover, we compared the performance of Laplacian Eigenmaps and Isomap embeddings on pose classification.[7] The data was randomly split into a training and a test set, and K-Nearest Neigh-

7. KNN only uses nearest neighbor information for classification. Since neighborhoods are considered to be locally linear in the input space, we expect KNN to perform well in the input space. Hence, using KNN to compare low-level embeddings indirectly measures how well nearest neighbor information is preserved.

bor (KNN) was used for classification. $K = 1$ gives lower error than higher K as shown in Table 3(a). Also, the classification error is lower for both exact and approximate Isomap than for Laplacian Eigenmaps, suggesting that neighborhood information is better preserved by Isomap (Table 3). Note that, similar to clustering, the Nyström approximation performs as well as Exact Isomap (Table 3(a)). Better clustering and classification results (along with superior 2D visualizations as shown in Appendix E), imply that approximate Isomap outperforms exact Laplacian Eigenmaps. Moreover, the Nyström approximation is computationally cheaper and empirically more effective than the Column sampling approximation. Thus, we used Nyström Isomap to generate embeddings for Webfaces-18M.

After learning a face manifold from Webfaces-18M, we analyzed the results with various visualizations. The top row of Figure 5 shows the 2D embeddings from Nyström Isomap. The top left figure shows the face samples from various locations in the manifold. It is interesting to see that embeddings tend to cluster the faces by pose. These results support the good clustering performance observed using Isomap on PIE data. Also, two groups (bottom left and top right) with similar poses but different illuminations are projected at different locations. Additionally, since 2D projections are very condensed for 18M points, one can expect more discrimination for higher k, for example, $k = 100$.

In Figure 5, the top right figure shows the shortest paths on the manifold between different public figures. The images along the corresponding paths have smooth transitions as shown in the bottom of the figure. In the limit of infinite samples, Isomap guarantees that the distance along the shortest path between any pair of points will be preserved as Euclidean distance in the embedded space. Even though the paths in the figure are reasonable approximations of straight lines in the embedded space, these results suggest that either (i) 18M faces are perhaps not enough samples to learn the face manifold exactly, or (ii) a low-dimensional manifold of faces may not actually exist (perhaps the data clusters into multiple low dimensional manifolds). It remains an open question as to how we can measure and evaluate these hypotheses, since even very large scale testing has not provided conclusive evidence.

5. Conclusion

We have studied sampling based low-rank approximation algorithms, presenting an analysis of two techniques for approximating SVD on large dense SPSD matrices and providing a theoretical and empirical comparison. Although the Column sampling method generates more accurate singular values, singular vectors and low-rank matrix projections, the Nyström method constructs better low-rank approximations, which are of great practical interest as they do not use the full matrix. Furthermore, our large-scale manifold learning studies illustrate the applicability of these algorithms when working with large dense kernel matrices, reveal that Isomap coupled with the Nyström approximation can effectively extract low-dimensional structure from data sets containing millions of images.

Appendix A. Proof of Theorem 1

Proof From (5), it is easy to see that

$$\widetilde{\mathbf{K}}_k^{col} = \mathbf{U}_{C,k}\mathbf{U}_{C,k}^\top\mathbf{K} = \mathbf{U}_C\mathbf{R}_{col}\mathbf{U}_C^\top\mathbf{K},$$

where $\mathbf{R}_{col} = \left[\begin{smallmatrix} \mathbf{I}_k & 0 \\ 0 & 0 \end{smallmatrix}\right]$. Similarly, from (6) we can derive

$$\widetilde{\mathbf{K}}_k^{nys} = \mathbf{U}_C \mathbf{R}_{nys} \mathbf{U}_C^\top \mathbf{K} \text{ where } \mathbf{R}_{nys} = \mathbf{Y}(\mathbf{\Sigma}_{W,k}^2)^+ \mathbf{Y}^\top,$$

and $\mathbf{Y} = \sqrt{l/n}\mathbf{\Sigma}_C \mathbf{V}_C^\top \mathbf{U}_{W,k}$. Note that both $\mathbf{R}_{col}$ and $\mathbf{R}_{nys}$ are SPSD matrices. Furthermore, if $k = l$, $\mathbf{R}_{col} = \mathbf{I}_l$. Let $\mathbf{E}$ be the (squared) reconstruction error for an approximation of the form $\mathbf{U}_C \mathbf{R} \mathbf{U}_C^\top \mathbf{K}$, where $\mathbf{R}$ is an arbitrary SPSD matrix. Hence, when $k = l$, the difference in reconstruction error between the generic and the Column sampling approximations is

$$\begin{aligned}
\mathbf{E} - \mathbf{E}_{col} =& \|\mathbf{K} - \mathbf{U}_C \mathbf{R} \mathbf{U}_C^\top \mathbf{K}\|_F^2 - \|\mathbf{K} - \mathbf{U}_C \mathbf{U}_C^\top \mathbf{K}\|_F^2 \\
=& \operatorname{Tr}\left[\mathbf{K}^\top (\mathbf{I}_n - \mathbf{U}_C \mathbf{R} \mathbf{U}_C^\top)^\top (\mathbf{I}_n - \mathbf{U}_C \mathbf{R} \mathbf{U}_C^\top) \mathbf{K}\right] \\
& - \operatorname{Tr}\left[\mathbf{K}^\top (\mathbf{I}_n - \mathbf{U}_C \mathbf{U}_C^\top)^\top (\mathbf{I}_n - \mathbf{U}_C \mathbf{U}_C^\top) \mathbf{K}\right] \\
=& \operatorname{Tr}\left[\mathbf{K}^\top (\mathbf{U}_C \mathbf{R}^2 \mathbf{U}_C^\top - 2\mathbf{U}_C \mathbf{R} \mathbf{U}_C^\top + \mathbf{U}_C \mathbf{U}_C^\top) \mathbf{K}\right] \\
=& \operatorname{Tr}\left[((\mathbf{R} - \mathbf{I}_n)\mathbf{U}_C^\top \mathbf{K})^\top ((\mathbf{R} - \mathbf{I}_n)\mathbf{U}_C^\top \mathbf{K})\right] \\
\geq& 0.
\end{aligned}$$

We used the facts that $\mathbf{U}_C^\top \mathbf{U}_C = \mathbf{I}_n$ and $\mathbf{A}^\top \mathbf{A}$ is SPSD for any matrix $\mathbf{A}$. ■

Appendix B. Proof of Theorem 2

Proof If $\alpha = \sqrt{n/l}$, then starting from (8) and expressing $\mathbf{C}$ and $\mathbf{W}$ in terms of $\mathbf{X}$ and $\mathbf{S}$, we have

$$\begin{aligned}
\widetilde{\mathbf{K}}_k^{col} =& \alpha \mathbf{K}\mathbf{S}((\mathbf{S}^\top \mathbf{K}^2 \mathbf{S})_k^{1/2})^+ \mathbf{S}^\top \mathbf{K}^\top \\
=& \alpha \mathbf{X}^\top \mathbf{X}' \left((\mathbf{V}_{C,k} \mathbf{\Sigma}_{C,k}^2 \mathbf{V}_{C,k}^\top)^{1/2}\right)^+ \mathbf{X}'^\top \mathbf{X} \\
=& \mathbf{X}^\top \mathbf{U}_{X',k} \mathbf{Z}_{col} \mathbf{U}_{X',k}^\top \mathbf{X},
\end{aligned}$$

where $\mathbf{Z}_{col} = \alpha \mathbf{\Sigma}_{X'} \mathbf{V}_{X'}^\top \mathbf{V}_{C,k} \mathbf{\Sigma}_{C,k}^+ \mathbf{V}_{C,k}^\top \mathbf{V}_{X'} \mathbf{\Sigma}_{X'}$. Similarly, from (7) we have:

$$\begin{aligned}
\widetilde{\mathbf{K}}_k^{nys} =& \mathbf{K}\mathbf{S}(\mathbf{S}^\top \mathbf{K}\mathbf{S})_k^+ \mathbf{S}^\top \mathbf{K}^\top \\
=& \mathbf{X}^\top \mathbf{X}' \left(\mathbf{X}'^\top \mathbf{X}'\right)_k^+ \mathbf{X}'^\top \mathbf{X} \\
=& \mathbf{X}^\top \mathbf{U}_{X',k} \mathbf{U}_{X',k}^\top \mathbf{X}.
\end{aligned} \tag{9}$$

Clearly, $\mathbf{Z}_{nys} = \mathbf{I}_k$. Next, we analyze the error, $\mathbf{E}$, for an arbitrary $\mathbf{Z}$, which yields the approximation $\widetilde{\mathbf{K}}_k^Z$:

$$\mathbf{E} = \|\mathbf{K} - \widetilde{\mathbf{K}}_k^Z\|_F^2 = \|\mathbf{X}^\top (\mathbf{I}_N - \mathbf{U}_{X',k} \mathbf{Z} \mathbf{U}_{X',k}^\top) \mathbf{X}\|_F^2.$$

Let $\mathbf{X} = \mathbf{U}_X \mathbf{\Sigma}_X \mathbf{V}_X^\top$ and $\mathbf{Y} = \mathbf{U}_X^\top \mathbf{U}_{X',k}$. Then,

$$\begin{aligned}
\mathbf{E} =& \operatorname{Tr}\left[\left(\mathbf{U}_X \mathbf{\Sigma}_X \mathbf{U}_X^\top (\mathbf{I}_N - \mathbf{U}_{X',k} \mathbf{Z} \mathbf{U}_{X',k}^\top) \mathbf{U}_X \mathbf{\Sigma}_X \mathbf{U}_X^\top\right)^2\right] \\
=& \operatorname{Tr}\left[\left(\mathbf{U}_X \mathbf{\Sigma}_X (\mathbf{I}_N - \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top) \mathbf{\Sigma}_X \mathbf{U}_X^\top\right)^2\right] \\
=& \operatorname{Tr}\left[\mathbf{\Sigma}_X (\mathbf{I}_N - \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top) \mathbf{\Sigma}_X^2 (\mathbf{I}_N - \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top) \mathbf{\Sigma}_X\right] \\
=& \operatorname{Tr}\left[\mathbf{\Sigma}_X^4 - 2\mathbf{\Sigma}_X^2 \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top \mathbf{\Sigma}_X^2 + \mathbf{\Sigma}_X \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top \mathbf{\Sigma}_X^2 \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top \mathbf{\Sigma}_X\right].
\end{aligned} \tag{10}$$

Figure 6: (a) A few random samples from the largest connected component of the Webfaces-18M neighborhood graph. (b) Visualization of disconnected components of the neighborhood graphs from Webfaces-18M (top row) and from PIE-35K (bottom row). The neighbors for each of these images are all within this set, thus making the entire set disconnected from the rest of the graph. Note that these images are not exactly the same. (c) Visualization of disconnected components containing exactly one image. Although several of the images above are not faces, some are actual faces, suggesting that certain areas of the face manifold are not adequately sampled by Webfaces-18M.

To find $\mathbf{Z}^*$, the $\mathbf{Z}$ that minimizes (10), we use the convexity of (10) and set:

$$\partial \mathbf{E}/\partial \mathbf{Z} = -2\mathbf{Y}^\top \mathbf{\Sigma}_X^4 \mathbf{Y} + 2(\mathbf{Y}^\top \mathbf{\Sigma}_X^2 \mathbf{Y})\mathbf{Z}^*(\mathbf{Y}^\top \mathbf{\Sigma}_X^2 \mathbf{Y}) = 0$$

and solve for $\mathbf{Z}^*$, which gives us:

$$\mathbf{Z}^* = (\mathbf{Y}^\top \mathbf{\Sigma}_X^2 \mathbf{Y})^+ (\mathbf{Y}^\top \mathbf{\Sigma}_X^4 \mathbf{Y})(\mathbf{Y}^\top \mathbf{\Sigma}_X^2 \mathbf{Y})^+.$$

$\mathbf{Z}^* = \mathbf{Z}_{nys} = \mathbf{I}_k$ if $\mathbf{Y} = \mathbf{I}_k$, though $\mathbf{Z}^*$ does not in general equal either $\mathbf{Z}_{col}$ or $\mathbf{Z}_{nys}$, which is clear by comparing the expressions of these three matrices.[8] Furthermore, since $\mathbf{\Sigma}_X^2 = \mathbf{\Sigma}_K$, $\mathbf{Z}^*$ depends on

8. This fact is illustrated in our experimental results for the 'DEXT' data set in Figure 2(b).

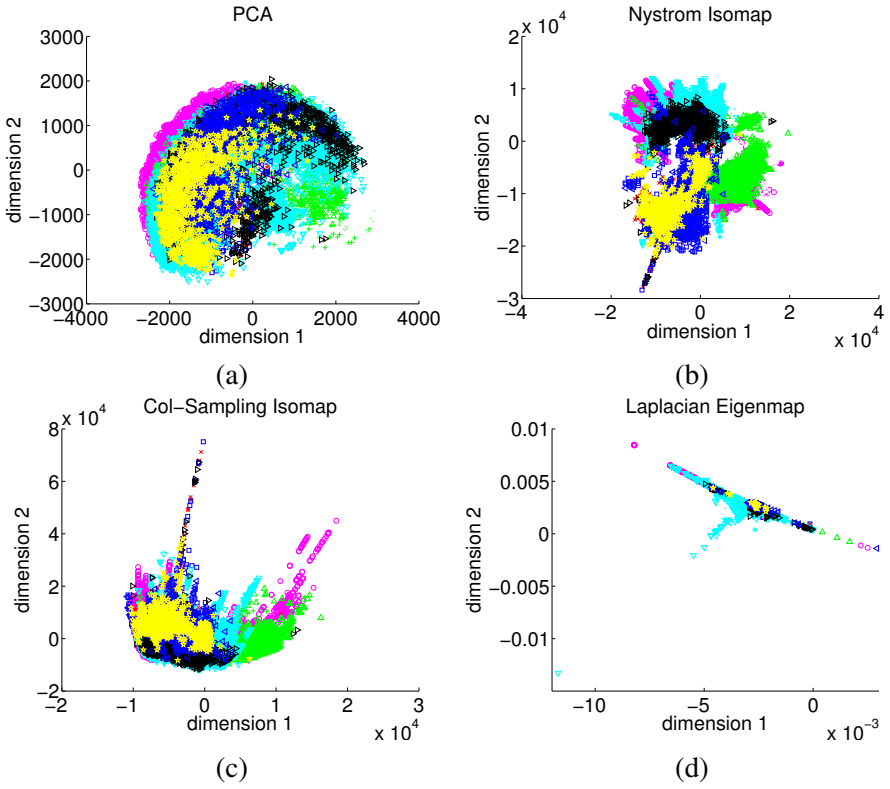


Figure 7: Optimal 2D projections of PIE-35K where each point is color coded according to its pose label. (a) PCA projections tend to spread the data to capture maximum variance. (b) Isomap projections with Nyström approximation tend to separate the clusters of different poses while keeping the cluster of each pose compact. (c) Isomap projections with Column sampling approximation have more overlap than with Nyström approximation. (d) Laplacian Eigenmaps projects the data into a very compact range.

the spectrum of $\mathbf{K}$. ■

Appendix C. Proof of Theorem 3

Proof Since $\mathbf{K} = \mathbf{X}^\top \mathbf{X}$, $\text{rank}(\mathbf{K}) = \text{rank}(\mathbf{X}) = r$. Similarly, $\mathbf{W} = \mathbf{X}'^\top \mathbf{X}'$ implies $\text{rank}(\mathbf{X}') = r$. Thus the columns of $\mathbf{X}'$ span the columns of $\mathbf{X}$ and $\mathbf{U}_{X',r}$ is an orthonormal basis for $\mathbf{X}$, that is, $\mathbf{I}_N - \mathbf{U}_{X',r}\mathbf{U}_{X',r}^\top \in \text{Null}(\mathbf{X})$. Since $k \geq r$, from (9) we have

$$\|\mathbf{K} - \widetilde{\mathbf{K}}_k^{nys}\|_F = \|\mathbf{X}^\top(\mathbf{I}_N - \mathbf{U}_{X',r}\mathbf{U}_{X',r}^\top)\mathbf{X}\|_F = 0,$$

which proves the first statement of the theorem. To prove the second statement, we note that $\text{rank}(C) = r$. Thus, $\mathbf{C} = \mathbf{U}_{C,r}\mathbf{\Sigma}_{C,r}\mathbf{V}_{C,r}^\top$ and $(\mathbf{C}^\top\mathbf{C})_k^{1/2} = (\mathbf{C}^\top\mathbf{C})^{1/2} = \mathbf{V}_{C,r}\mathbf{\Sigma}_{C,r}\mathbf{V}_{C,r}^\top$ since $k \geq r$. If $\mathbf{W} = (1/\alpha)(\mathbf{C}^\top\mathbf{C})^{1/2}$, then the Column sampling and Nyström approximations are identical and

hence exact. Conversely, to exactly reconstruct $\mathbf{K}$, Column sampling necessarily reconstructs $\mathbf{C}$ exactly. Using $\mathbf{C}^\top = [\mathbf{W} \quad \mathbf{K}_{21}^\top]$ in (8) we have:

$$\begin{aligned}
\widetilde{\mathbf{K}}_k^{col} = \mathbf{K} \quad &\Longrightarrow \quad \alpha \mathbf{C} \big((\mathbf{C}^\top \mathbf{C})_k^{\frac{1}{2}} \big)^+ \mathbf{W} = \mathbf{C} \\
&\Longrightarrow \quad \alpha \mathbf{U}_{C,r} \mathbf{V}_{C,r}^\top \mathbf{W} = \mathbf{U}_{C,r} \mathbf{\Sigma}_{C,r} \mathbf{V}_{C,r}^\top \\
&\Longrightarrow \quad \alpha \mathbf{V}_{C,r} \mathbf{V}_{C,r}^\top \mathbf{W} = \mathbf{V}_{C,r} \mathbf{\Sigma}_{C,r} \mathbf{V}_{C,r}^\top \qquad (11) \\
&\Longrightarrow \quad \mathbf{W} = \frac{1}{\alpha} (\mathbf{C}^\top \mathbf{C})^{1/2}. \qquad (12)
\end{aligned}$$

In (11) we use $\mathbf{U}_{C,r}^\top \mathbf{U}_{C,r} = \mathbf{I}_r$, while (12) follows since $\mathbf{V}_{C,r} \mathbf{V}_{C,r}^\top$ is an orthogonal projection onto the span of the rows of $\mathbf{C}$ and the columns of $\mathbf{W}$ lie within this span implying $\mathbf{V}_{C,r} \mathbf{V}_{C,r}^\top \mathbf{W} = \mathbf{W}$. ■

Appendix D. Visualization of Connected Components in Neighborhood Graph

Figure 6(a) shows a few random samples from the largest component of the neighborhood graph we generate for Webfaces-18M. Images not within the largest component are either part of a strongly connected set of images (Figure 6(b)) or do not have any neighbors within the upper distance limit (Figure 6(c)). There are significantly more false positives in Figure 6(c) than in Figure 6(a), although some of the images in Figure 6(c) are actually faces. Clearly, the distance limit introduces a trade-off between filtering out non-faces and excluding actual faces from the largest component.

Appendix E. Visualization of 2D Embeddings of PIE-35K

Figure 7 shows the optimal 2D projections from different methods for PIE-35K.

References

C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. OpenFST: A general and efficient weighted finite-state transducer library. In *Conference on Implementation and Application of Automata*, 2007.

M. Balasubramanian and E. L. Schwartz. The Isomap algorithm and topological stability. *Science*, 295, 2002.

M. Belkin and P. Niyogi. Laplacian Eigenmaps and spectral techniques for embedding and clustering. In *Neural Information Processing Systems*, 2001.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Conference on Learning Theory*, 1992.

E. Chang, K. Zhu, H. Wang, H. Bai, J. Li, Z. Qiu, and H. Cui. Parallelizing support vector machines on distributed computers. In *Neural Information Processing Systems*, 2008.

C. Cortes, M. Mohri, and A. Talwalkar. On the impact of kernel approximation on learning accuracy. In *Conference on Artificial Intelligence and Statistics*, 2010.

V. de Silva and J. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Neural Information Processing Systems*, 2003.

A. Deshpande, L. Rademacher, S. Vempala, and G. Wang. Matrix approximation and projective clustering via volume sampling. In *Symposium on Discrete Algorithms*, 2006.

I. Dhillon and B. Parlett. Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices. *Linear Algebra and its Applications*, 387:1–28, 2004.

D. Donoho and C. Grimes. Hessian Eigenmaps: locally linear embedding techniques for high dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596, 2003.

P. Drineas and M. W. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1), 2006.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2002.

C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

A. Frieze, R. Kannan, and S. Vempala. Fast Monte Carlo algorithms for finding low-rank approximations. In *Foundation of Computer Science*, 1998.

G. Golub and C. V. Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 2nd edition, 1983. ISBN 0-8018-3772-3 (hardcover), 0-8018-3739-1 (paperback).

N. Halko, P. Martinsson, and J. Tropp. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. `arXiv:0909.4061v1[math.NA]`, 2009.

J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *International Conference on Machine Learning*, 2004.

X. He, S. Yan, Y. Hu, and P. Niyogi. Face recognition using Laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

T. Joachims. Making large-scale support vector machine learning practical. In *Neural Information Processing Systems*, 1999.

S. Kumar and H. Rowley. People Hopper. `http://googleresearch.blogspot.com/2010/03/hopping-on-face-manifold-via-people.html`, 2010.

S. Kumar, M. Mohri, and A. Talwalkar. On sampling-based approximate spectral decomposition. In *International Conference on Machine Learning*, 2009.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling methods for the Nyström method. In *Journal of Machine Learning Research*, 2012.

Y. LeCun and C. Cortes. The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/, 1998.

T. Liu, A. W. Moore, A. G. Gray, and K. Yang. An investigation of practical approximate nearest neighbor algorithms. In *Neural Information Processing Systems*, 2004.

L. Mackey, A. Talwalkar, and M. I. Jordan. Divide-and-conquer matrix factorization. In *Neural Information Processing Systems*, 2011.

M. W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

G. Mann, R. McDonald, M. Mohri, N. Silberman, and D. Walker. Efficient large-scale distributed training of conditional maximum entropy models. In *Neural Information Processing Systems*, 2009.

E. Nyström. Über die praktische auflösung von linearen integralgleichungen mit anwendungen auf randwertaufgaben der potentialtheorie. *Commentationes Physico-Mathematicae*, 4(15):1–52, 1928.

J. Platt. Fast training of Support Vector Machines using sequential minimal optimization. In *Neural Information Processing Systems*, 1999.

J. Platt. Fast embedding of sparse similarity graphs. In *Neural Information Processing Systems*, 2004.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by Locally Linear Embedding. *Science*, 290(5500), 2000.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press: Cambridge, MA, 2002.

B. Shaw and T. Jebara. Structure preserving embedding. In *International Conference on Machine Learning*, 2009.

T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination, and expression database. In *Conference on Automatic Face and Gesture Recognition*, 2002.

A. Talwalkar. *Matrix Approximation for Large-scale Learning*. Ph.D. thesis, Computer Science Department, Courant Institute, New York University, New York, NY, 2010.

A. Talwalkar and A. Rostamizadeh. Matrix coherence and the Nyström method. In *Conference on Uncertainty in Artificial Intelligence*, 2010.

A. Talwalkar, S. Kumar, and H. Rowley. Large-scale manifold learning. In *Conference on Vision and Pattern Recognition*, 2008.

J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500), 2000.

K. Weinberger and L. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *AAAI Conference on Artificial Intelligence*, 2006.

C. K. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Neural Information Processing Systems*, 2000.

Journal of Machine Learning Research 14 (2013) 3153-3157 Submitted 3/11; Revised 4/13; Published 10/13

QuantMiner for Mining Quantitative Association Rules

Ansaf Salleb-Aouissi ANSAF@CCLS.COLUMBIA.EDU
Center for Computational Learning Systems (CCLS)
Columbia University
475 Riverside Drive, New York, NY 10115, USA

Christel Vrain CHRISTEL.VRAIN@UNIV-ORLEANS.FR
Cyril Nortet CYRIL.NORTET@UNIV-ORLEANS.FR
Laboratoire d'Informatique Fondamentale d'Orléans (LIFO)
Université d'Orléans
BP 6759, 45067 Orléans Cedex 2, France

Xiangrong Kong SHARONXKONG@GMAIL.COM
Vivek Rathod VIVEKMRATHOD@GMAIL.COM
Center for Computational Learning Systems (CCLS)
Columbia University
475 Riverside Drive, New York, NY 10115, USA

Daniel Cassard D.CASSARD@BRGM.FR
French Geological Survey (BRGM)
3, avenue Claude Guillemin
BP 6009, Orléans Cedex 2, France

Editor: Mikio Braun

Abstract

In this paper, we propose QUANTMINER, a mining quantitative association rules system. This system is based on a genetic algorithm that dynamically discovers "good" intervals in association rules by optimizing both the support and the confidence. The experiments on real and artificial databases have shown the usefulness of QUANTMINER as an interactive, exploratory data mining tool.

Keywords: association rules, numerical and categorical attributes, unsupervised discretization, genetic algorithm, simulated annealing

1. Introduction

In this paper, we propose a software for mining quantitative and categorical rules, that implements the work proposed in Salleb-Aouissi et al. (2007). Given a set of categorical and quantitative attributes and a data set, the aim is to find rules built on these attributes that optimize given criteria. Expressions occurring in the rules are either $A = v$ for a categorical attribute, or $A \in [l,u]$ for a quantitative one. The main difficulty is to find the best bounds l and u of the intervals. Mining quantitative association rules cannot be considered as a direct extension of mining categorical rules. While this task has received less attention than mining Boolean association rules (Agrawal et al., 1993), it remains a very important one from the point of view of applications. Several approaches have been designed for this task. For instance, a preprocessing step, discretizing (also called binning)

numeric attributes into intervals is performed before the mining task in Srikant and Agrawal (1996). Some approaches (e.g., Aumann and Lindell, 1999) restrict learning to special kind of rules: the right-hand side of a rule expresses the distribution (e.g., mean, variance) of numeric attributes, the left-hand side is composed either of a set of categorical attributes or of a *single* discretized numeric attribute. Optimization-based approaches handle numeric attributes during the mining process. The first one proposed in Fukuda et al. (1996) introduces a new optimization criterion, called the *gain*, taking into account both the support and the confidence of a rule. Extensions have been proposed in Brin et al. (2003) but the forms of the rules remain restricted to one or two numeric attributes. A genetic algorithm is also proposed in Mata et al. (2002) to optimize the support of itemsets defined on uninstantiated intervals of numeric attributes. This approach is limited to numeric attributes and optimizes only the support before mining association rules. QuantMiner handles both categorical and numerical attributes and optimizes the intervals of numeric attributes during the process of mining association rules. It is based on a genetic algorithm, the fitness function aims at maximizing the gain of an association rule while penalizing the attributes with large intervals. Recent work (e.g., Alcalá-Fdez et al., 2010) show the interest of evolutionary algorithms for such a task.

2. QuantMiner

In the following, an item is either an expression $A = v$, where A is a categorical (also called qualitative) attribute and v is a value from its domain, or an expression $A \in [l,u]$ where A is a numerical (also called quantitative) attribute. QuantMiner optimizes a set of rule patterns produced from a user-specified rule template.

Rule templates: A *rule template* is a preset format of a quantitative association rule used as a starting point for the quantitative mining process. It is defined by the set of attributes occurring in the left hand side and the right hand side or both sides of the rule. Categorical attributes may or may not have specific values. Furthermore, an attribute may be mandatory or optional, thus allowing to generate rules of different lengths from the same rule template. Given a rule template, first for each unspecified categorical attribute in the template, the frequent values are computed. Then, a set of *rule patterns* verifying the specifications (position of the attributes, mandatory/optional presence of the attributes, values for categorical attributes either provided, or computed as frequent) are built. For each rule pattern, the algorithm looks for the "best" intervals for the numeric attributes occurring in that template, relying on a genetic algorithm.

Example 1 *Consider the Iris data set from the UCI machine learning repository.*[1] *An example of rule template and a specific example of rule are given in Figure 1 and in Figure 2 respectively.*

Population: An individual is a set of items of the form $attribute_i \in [l_i,u_i]$, where $attribute_i$ is the i^{th} numeric attribute in the rule template from the left to the right. The process for generating the population is described in Salleb-Aouissi et al. (2007).

Genetic operators: Mutation and *crossover* are both used in QuantMiner. For the crossover operator, for each attribute the interval is either inherited from one of the parents or formed by mixing the bounds of the two parents. For an individual, mutation increases or decreases the lower or upper bound of its intervals. Moving interval bounds is done so as to discard/involve no more than 10% of tuples already covered by the interval.

1. The data set is available here: `http://archive.ics.uci.edu/ml/datasets/Iris`.

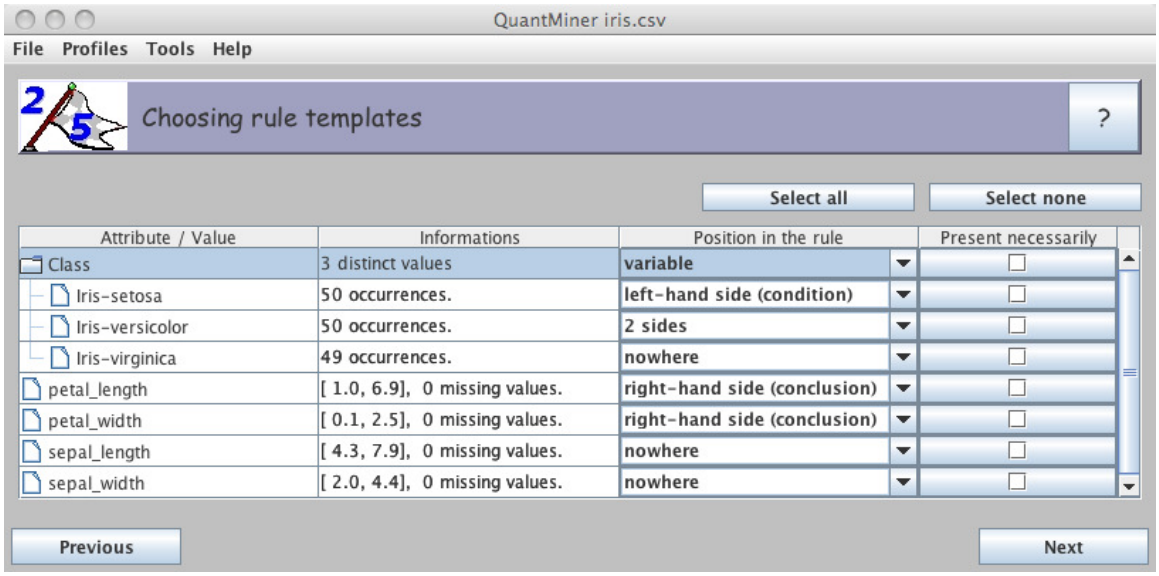


Figure 1: An example of rule template exploring petal attributes for 2 categories of iris. *petal_length* and *petal_width* are chosen to be on the right side of the rules template.

Fitness function: It is based on the *Gain* proposed in Fukuda et al. (1996), defined in Equation 1:

$$Gain(A \Rightarrow B) = Supp(A \wedge B) - MinConf * Supp(A). \quad (1)$$

Let $\mathcal{A}_{num}$ the set of numerical attributes present in the rule $A \Rightarrow B$. Let I_a denote the interval of the attribute $a \in \mathcal{A}_{num}$. In the following, $size(a)$ denotes the length of the smallest interval which contains all the data for the attribute a and $size(I_a)$ denotes the length of the interval I_a. If $Gain(A \Rightarrow B)$ is negative, then the fitness of the rule is set to the gain. If it is positive (the confidence of the rule exceeds the minimum confidence threshold), the proportions of the intervals (defined as the ratios between the sizes and the domains) is taken into account, so as to favor those with small sizes as shown in Equation 2. Moreover, rules with low supports are penalized by decreasing drastically their fitness values.

$$Fitness(A \Rightarrow B) = Gain(A \Rightarrow B) \times \prod_{a \in \mathcal{A}_{num}} \left(1 - \frac{size(I_a)}{size(a)}\right)^2. \quad (2)$$

3. Implementation

We developed QUANTMINER in JAVA as a 5-step GUI wizard allowing an interactive mining process.[2] After opening a data set, the user can choose attributes, a rule template, the optimization technique and set its parameters, launch the process, and finally display the rules with various sorting options: support, confidence or rule length. The user can save the mining-context, go back to

2. The software is available at `http://quantminer.github.com/QuantMiner/`.

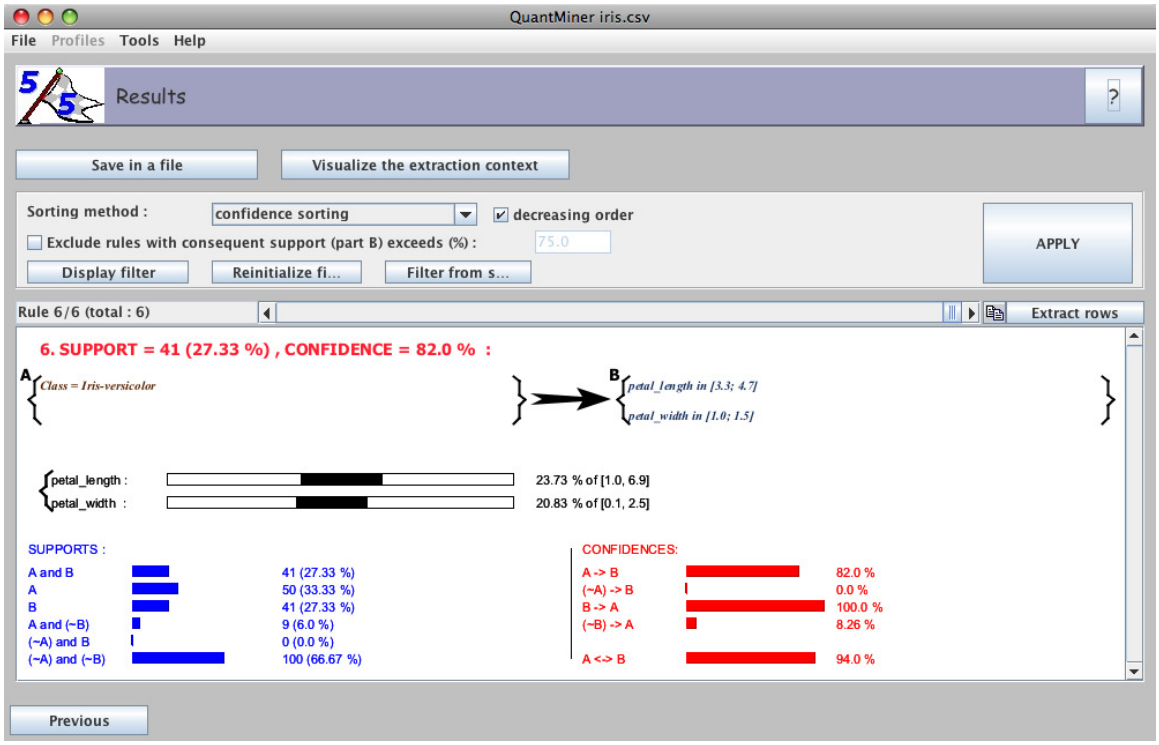


Figure 2: Example of rule visualization in QuantMiner. The top part shows filtering criteria that facilitate exploring the rules. The bottom part shows a specific rule followed by the proportion of each interval in its corresponding domain. More measures are given to assess the quality of the rule, for example, $confidence(\neg A \Rightarrow B)$.

previous steps, change the method, parameters, templates and restart the learning. Note that simulated annealing is implemented in QuantMiner as an alternative optimization method. A tentative for mining rules with disjunctive intervals is also implemented. We hope this functionality will be further investigated.

Acknowledgments

This project has been supported by the BRGM-French Geological Survey, LIFO-Université d'Orléans and CCLS-Columbia University. Many thanks to everyone who contributed to QuantMiner with support, ideas, data and feedback.

References

R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD*, pages 207–216, 1993.

J. Alcalá-Fdez, N. Flugy Papè, A. Bonarini, and F. Herrera. Analysis of the effectiveness of the genetic algorithms based on extraction of association rules. *Fundam. Inform.*, 98(1):1–14, 2010.

Y. Aumann and Y. Lindell. A statistical theory for quantitative association rules. In *Knowledge Discovery and Data Mining*, pages 261–270, 1999.

S. Brin, R. Rastogi, and K. Shim. Mining optimized gain rules for numeric attributes. *IEEE Trans. Knowl. Data Eng.*, 15(2):324–338, 2003.

T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Mining optimized association rules for numeric attributes. In *Proc. of the fteenth ACM SIGACTSIGMOD -SIGART PODS'96*, pages 182–191. ACM Press, 1996.

J. Mata, J. L. Alvarez, and J. C. Riquelme. An evolutionary algorithm to discover numeric association rules. In *Proceedings of the ACM SAC'2002*, pages 590–594, 2002.

A. Salleb-Aouissi, C. Vrain, and C. Nortet. Quantminer: A genetic algorithm for mining quantitative association rules. In *IJCAI*, pages 1035–1040, 2007.

R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proc. of the ACM SIGMOD*, pages 1–12, 1996.

Journal of Machine Learning Research 14 (2013) 3159-3163 Submitted 6/13; Revised 8/13; Published 10/13

Divvy: Fast and Intuitive Exploratory Data Analysis

Joshua M. Lewis JOSH@COGSCI.UCSD.EDU
Virginia R. de Sa DESA@COGSCI.UCSD.EDU
Department of Cognitive Science
University of California, San Diego
La Jolla, CA 92093-0515, USA

Laurens van der Maaten LVDMAATEN@GMAIL.COM
Faculty of Elect. Eng., Math., and Comp. Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Editor: Mark Reid

Abstract

Divvy is an application for applying unsupervised machine learning techniques (clustering and dimensionality reduction) to the data analysis process. Divvy provides a novel UI that allows researchers to tighten the action-perception loop of changing algorithm parameters and seeing a visualization of the result. Machine learning researchers can use Divvy to publish easy to use reference implementations of their algorithms, which helps the machine learning field have a greater impact on research practices elsewhere.

Keywords: clustering, dimensionality reduction, open source software, human computer interaction, data visualization

1. Introduction

The field of machine learning has produced many techniques for performing data analysis, but researchers outside the field face substantial challenges applying them to their data. First, new techniques are difficult to access. Authors often describe an algorithm without providing a reference implementation, and if they do provide code it may be for an unfamiliar language or platform. Second, new techniques are difficult to apply correctly. A new technique might make strong assumptions about the structure of its input or be very sensitive to parameter changes. Third, in the early, exploratory stage of data analysis researchers should explore and compare several different techniques. If each technique is challenging to get running for the reasons above, the challenge is compounded with multiple techniques using different languages and data formats.

We have built Divvy to ameliorate these problems for those who would like to use unsupervised machine learning techniques in their research (specifically clustering and dimensionality reduction), and provide a platform for machine learning researchers to publish fast, easy to use versions of their algorithms. Using Divvy, researchers can quickly run an assortment of clustering and dimensionality reduction algorithms on their data, without worrying about programming languages, data formats, or visualization.

Divvy is a member of a family of attempts to bring machine learning and data visualization to a wider audience. The GGobi project (Swayne et al., 2003) provides a variety of interactive visualiza-

tions for high-dimensional data, and includes some lightweight machine learning components such as PCA. The Orange project (University of Ljubljana Bioinformatics Laboratory, 2013) provides a visual programming interface for machine learning techniques in Python. In the commercial sector, Ayasdi, Inc. uses topology to help customers analyze their data intuitively (Ayasdi, Inc., 2013). Divvy's unique focus among these projects is on user experience and extensibility.

For users, Divvy provides a simple, fast interface for doing data analysis. For machine learning researchers, Divvy provides a plugin architecture that lets one release a version of an algorithm complete with custom UI and help resources. By publishing an algorithm on the Divvy platform, machine learning researchers can drastically lower the barriers to entry that data analysts face when attempting to use it. Divvy and all of its included plugins are open source, distributed under the MIT license.

2. Interaction

Divvy has three fundamental components to its interface (see Figure 1). In the lower-right corner of the main window (Label 1) is a collection of data sets for the user to analyze. Each data set is associated with one or more data set views, which are visualized in the left hand portion of the interface (Label 2). Finally each data set view is characterized by a combination of four plugins from the top-right panel (Label 3): a dimensionality reduction algorithm, a clustering algorithm, a point visualizer, and a data set visualizer.

Clustering and dimensionality reduction plugins are interfaces to their associated algorithms, currently K-means and single/complete linkage for clustering, and PCA, Isomap (Tenenbaum et al., 2000) and t-SNE (van der Maaten and Hinton, 2008) for dimensionality reduction. Point visualizers render data points in a meaningful way, for example, by rendering images in a data set where points represent images, or rendering type for a linguistic corpus. Data set visualizers render the entire data set, for example as a scatter plot or parallel coordinates plot. Divvy includes an image point visualizer and a scatter plot data set visualizer by default.

As an example, the user in Figure 1 has loaded four data sets into Divvy and selected the faces data set. The user has created three views that represent three distinct perspectives on the data. The top-right view is an Isomap embedding of the data set with an image point visualizer, so the user sees a selection of points rendered as images and positioned in the first two Isomap dimensions. The lower left view is a t-SNE embedding with coloring from K-means clustering rendered as a scatter plot.

In practice a researcher can use these visualizations to evaluate different approaches to dimensionality reduction and clustering. In Figure 1 the user can easily see that Isomap embeds the face images in two dimension more smoothly than t-SNE. The structure of the faces data set is ideal for Isomap, and a researcher can quickly discover that with Divvy.

Users can create as many perspectives on their data as they'd like, grow or shrink them with the slider on the lower-right edge of the screen, and export their preferred views as PNGs. Whenever they change a plugin or plugin parameter, the selected view automatically rerenders with the new setting. In order to make Divvy as responsive as possible, each data set view is sandboxed in its own set of threads (Divvy is task parallel with granularity at the view level). Users can start a long computation in one view while still interacting with other views or data sets. For shorter computations the view changes instantly, giving users immediate visual feedback on the effect of their parameter selections.

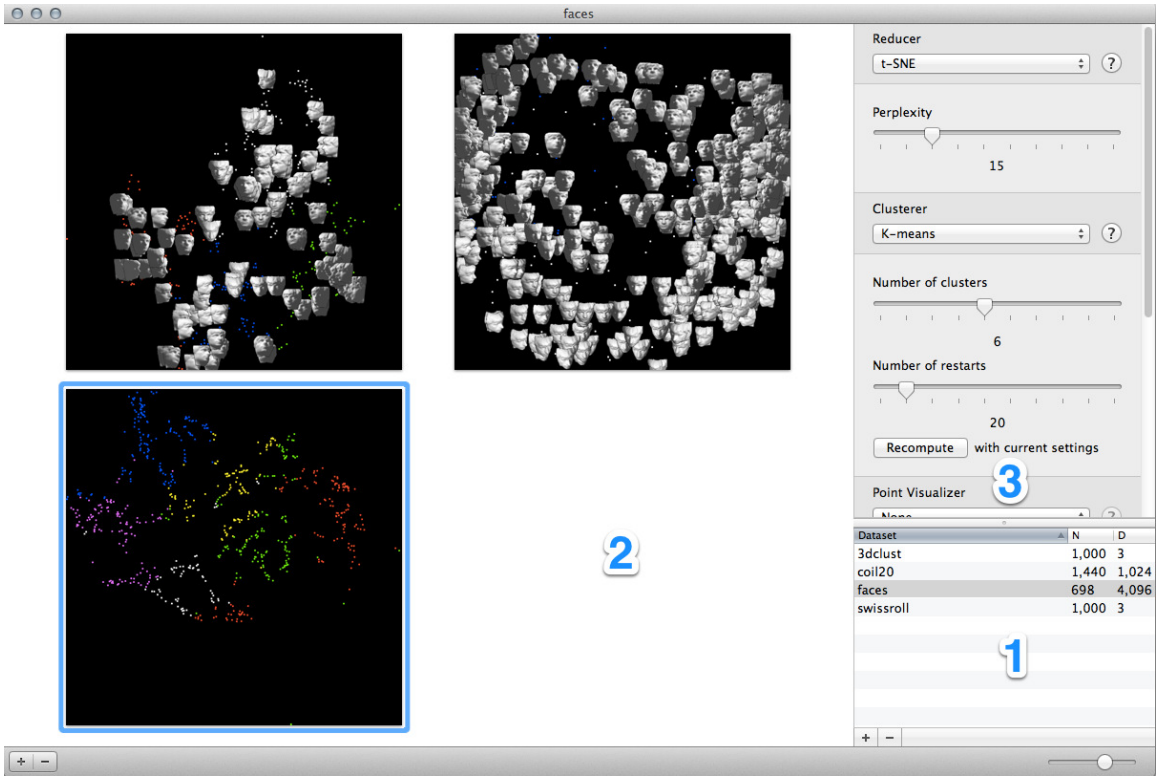


Figure 1: Divvy's UI. (1) The data sets list—data sets the user has loaded appear here with some summary statistics. (2) The data set view palette—data set views are visualizations of data sets using a combination of a dimensionality reduction algorithm, a clustering algorithm, a point visualizer, and a data set visualizer. Each data set can have multiple data set views. (3) Data set view parameters—controls for setting the parameters of the algorithms that compose a particular data set view, for example, the number of clusters for a clustering algorithm.

Our goal with the Divvy UI is to tighten up the action-perception loop in data analysis. As an analogy, baseball players have an excellent idea of how baseballs behave. A baseball's behavior is, of course, governed by the laws of physics and an explicit description of that behavior might be quite complex when spin, deformation, wind and field texture are taken into account. Nevertheless, through extensive experience baseball players acquire an excellent pragmatic understanding of how baseballs behave, an understanding that one might guess is based on an implicit learned model of baseball behavior rather than the explicit model a physicist would give. By giving Divvy users an immediate, tactile experience of algorithmic behavior, we hope they can develop better intuitive models of how algorithms behave and thus make better analysis decisions.

3. Architecture

The core Divvy application performs no computation. Rather it is a lightweight framework for loading data sets and plugins and then coordinating their interaction. Each plugin is an independent bundle that defines a UI and follows one of four possible input/output protocols: clusterer, reducer, point visualizer and data set visualizer.

While the core Divvy application is Mac OS X specific, each machine learning plugin is just a lightly-wrapped reference implementation of its algorithm in C or C++.[1] A researcher publishing an algorithm in Divvy's format need only add an OS X UI (and Divvy gives complete freedom as to the details of that UI save a fixed width) to their implementation, which remains completely platform-agnostic. With this bit of work users can drop the algorithm bundle into Divvy and start using the new technique on their existing data sets.

We implemented Divvy on Mac OS X in order to focus on one user experience. While it's of course desirable to have open source software on as many platforms as possible, building the Divvy UI across platforms was outside the scope of our engineering resources.[2]

Divvy can import data from CSV or a simple BIN format, and we have released R to Divvy and MATLAB to Divvy exporters. Data within Divvy can be exported as PNG or CSV as appropriate. Our goal is that Divvy can fit well into diverse analysis workflows.

4. Performance

Divvy is both task and data parallel. As mentioned above, each data set view owns a set of threads that compute independently from the UI and those of other views. Within a view each plugin can operate in parallel over its assigned data set. In our lab Divvy can achieve over 2,300% CPU utilization on our hyperthreaded 12-core Mac Pro through a combination of these two forms of parallelization.[3]

The task parallelism is achieved at the application level through the NSOperation framework in Cocoa. Plugin authors get it for free. Data parallelism is the responsibility of plugin authors and should be implemented using the open-source libdispatch library.[4]

Acknowledgments

We would like to acknowledge support for this project from the National Science Foundation (NSF grant SES-0963071).

References

Ayasdi, Inc. Iris: Query-free insight discovery, 2013. URL `http://www.ayasdi.com/product/`.

1. Overall the Divvy codebase is two-thirds Objective-C (interface, plugin wrappers) and one-third C/C++ (algorithms). As algorithms are added, this balance will shift towards platform-independent code. To port Divvy to another platform, one would only need to rewrite the interface code.
2. Joshua Lewis wrote the entire Divvy application and bundled plugins save the dimensionality reduction plugins, which were written by Laurens van der Maaten.
3. With twelve cores and two threads per core, peak utilization is 2,400%.
4. We picked libdispatch because OpenMP has an issue with GCC 4.2/4.3 where starting a parallel section from a thread other than the main thread causes a crash, so we cannot recommend it.

Deborah F. Swayne, Duncan Temple Lang, Andreas Buja, and Dianne Cook. GGobi: evolving from XGobi into an extensible framework for interactive data visualization. *Computational Statistics & Data Analysis*, 43:423–444, 2003.

Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000.

University of Ljubljana Bioinformatics Laboratory. Orange - data mining fruitful and fun, 2013. URL `http://orange.biolab.si`.

Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, November 2008.

Journal of Machine Learning Research 14 (2013) 3165-3200 Submitted 2/13; Revised 7/13; Published 10/13

Variational Algorithms for Marginal MAP

Qiang Liu QLIU1@UCI.EDU
Alexander Ihler IHLER@ICS.UCI.EDU
Donald Bren School of Information and Computer Sciences
University of California, Irvine
Irvine, CA, 92697-3425, USA

Editor: Amir Globerson

Abstract

The marginal maximum *a posteriori* probability (MAP) estimation problem, which calculates the mode of the marginal posterior distribution of a subset of variables with the remaining variables marginalized, is an important inference problem in many models, such as those with hidden variables or uncertain parameters. Unfortunately, marginal MAP can be NP-hard even on trees, and has attracted less attention in the literature compared to the joint MAP (maximization) and marginalization problems. We derive a general dual representation for marginal MAP that naturally integrates the marginalization and maximization operations into a joint variational optimization problem, making it possible to easily extend most or all variational-based algorithms to marginal MAP. In particular, we derive a set of "mixed-product" message passing algorithms for marginal MAP, whose form is a hybrid of max-product, sum-product and a novel "argmax-product" message updates. We also derive a class of convergent algorithms based on proximal point methods, including one that transforms the marginal MAP problem into a sequence of standard marginalization problems. Theoretically, we provide guarantees under which our algorithms give globally or locally optimal solutions, and provide novel upper bounds on the optimal objectives. Empirically, we demonstrate that our algorithms significantly outperform the existing approaches, including a state-of-the-art algorithm based on local search methods.

Keywords: graphical models, message passing, belief propagation, variational methods, maximum *a posteriori*, marginal-MAP, hidden variable models

1. Introduction

Graphical models such as Bayesian networks and Markov random fields provide a powerful framework for reasoning about conditional dependency structures over many variables, and have found wide application in many areas including error correcting codes, computer vision, and computational biology (Wainwright and Jordan, 2008; Koller and Friedman, 2009). Given a graphical model, which may be estimated from empirical data or constructed by domain expertise, the term *inference* refers generically to answering probabilistic queries about the model, such as computing marginal probabilities or maximum *a posteriori* estimates. Although these inference tasks are NP-hard in the worst case, recent algorithmic advances, including the development of variational methods and the family of algorithms collectively called belief propagation, provide approximate or exact solutions for these problems in many practical circumstances.

In this work we will focus on three common types of inference tasks. The first involves *maximization* or *max-inference* tasks, sometimes called maximum *a posteriori* (MAP) or most probable

explanation (MPE) tasks, which look for a mode of the joint probability. The second are *sum-inference* tasks, which include calculating the marginal probabilities or the normalization constant of the distribution (corresponding to the probability of evidence in a Bayesian network). Finally, the main focus of this work is on *marginal MAP*, a type of *mixed-inference* problem that seeks a partial configuration of variables that maximizes those variables' marginal probability, with the remaining variables summed out.[1] Marginal MAP plays an essential role in many practical scenarios where there exist hidden variables or uncertain parameters. For example, a marginal MAP problem can arise as a MAP problem on models with hidden variables whose predictions are not of interest, or as a robust optimization variant of MAP with some unknown or noisily observed parameters marginalized w.r.t. a prior distribution. It can be also treated as a special case of the more complicated frameworks of stochastic programming (Birge and Louveaux, 1997) or decision networks (Howard and Matheson, 2005; Liu and Ihler, 2012).

These three types of inference tasks are listed in order of increasing difficulty: max-inference is NP-complete, while sum-inference is #P-complete, and mixed-inference is NP^{PP}-complete (Park and Darwiche, 2004; De Campos, 2011). Practically speaking, max-inference tasks have a host of efficient algorithms such as loopy max-product BP, tree-reweighted BP, and dual decomposition (see, e.g., Koller and Friedman, 2009; Sontag et al., 2011). Sum-inference is more difficult than max-inference: for example there are models, such as those with binary attractive pairwise potentials, on which sum-inference is #P-complete but max-inference is tractable (Greig et al., 1989; Jerrum and Sinclair, 1993).

Mixed-inference is even much harder than either max- or sum- inference problems alone: marginal MAP can be NP-hard even on tree structured graphs, as illustrated in the example by Koller and Friedman (2009) in Figure 1. The difficulty arises in part because the max and sum operators do not commute, causing the feasible elimination orders to have much higher induced width than for sum- or max-inference. Viewed another way, the marginalization step may destroy the dependency structure of the original graphical model, making the subsequent maximization step far more challenging. Probably for these reasons, there is much less work on marginal MAP than that on joint MAP or marginalization, despite its importance to many practical problems. In practice, it is common to over-use the simpler joint MAP or marginalization even when marginal MAP would be more appropriate. This may cause serious problems, as we illustrate in Example 1 and our empirical results in Section 9.

1.1 Contributions

We reformulate the mixed-inference problem to a joint maximization problem as a free energy objective that extends the well-known log-partition function duality form, making it possible to easily extend essentially arbitrary variational algorithms to marginal MAP. In particular, we propose a novel "mixed-product" BP algorithm that is a hybrid of max-product, sum-product, and a special "argmax-product" message updates, as well as a convergent proximal point algorithm that works by iteratively solving pure (or annealed) marginalization tasks. We also present junction graph BP variants of our algorithms, that work on models with higher order cliques. We also discuss mean field methods and highlight their connection to the expectation-maximization (EM) algorithm. We give theoretical guarantees on the global and local optimality of our algorithms for cases when the

1. In some literature (e.g., Park and Darwiche, 2004), marginal MAP is simply referred to as MAP, and the joint MAP problem is called MPE.

sum variables form tree structured subgraphs. Our numerical experiments show that our methods can provide significantly better solutions than existing algorithms, including a similar hybrid message passing algorithm by Jiang et al. (2011) and a state-of-the-art algorithm based on local search methods. A preliminary version of this work has appeared in Liu and Ihler (2011b).

1.2 Related Work

Expectation-maximization (EM) or variational EM provide one straightforward approach for marginal MAP, by viewing the sum nodes as hidden variables and the max nodes as parameters to be estimated; however, EM is prone to getting stuck at sub-optimal configurations. We show that EM can be treated as a special case of our framework when a mean field-like approximation is applied. Other classical state-of-the-art approaches include local search methods (e.g., Park and Darwiche, 2004), Markov chain Monte Carlo methods (e.g., Doucet et al., 2002; Yuan et al., 2004), and variational elimination based methods (e.g., Dechter and Rish, 2003; Mauá and de Campos, 2012). Jiang et al. (2011) recently proposed a hybrid message passing algorithm that has a similar form to our mixed-product BP algorithm, but without theoretical guarantees; we show in Section 5.3 that Jiang et al. (2011) can be viewed as an approximation of the marginal MAP problem that exchanges the order of sum and max operators. Another message-passing-style algorithm was proposed very recently in Altarelli et al. (2011) for general multi-stage stochastic optimization problems based on survey propagation, which again does not have optimality guarantees and has a relatively more complicated form. Finally, Ibrahimi et al. (2011) introduces a robust max-product belief propagation for solving a related worst-case robust optimization problem, where the hidden variables are minimized instead of marginalized. To the best of our knowledge, our work is the first general variational framework for marginal MAP, and provides the first strong optimality guarantees.

We begin in Section 2 by introducing background information on graphical models and variational inference. We then introduce a novel variational dual representation for marginal MAP in Section 3, and propose analogues of the Bethe and tree-reweighted approximations for marginal MAP in Section 4. A class of "mixed-product" message passing algorithms is proposed and analyzed in Section 5 and convergent alternatives are proposed in Section 6 based on proximal point methods. We then discuss the EM algorithm and its connection to our framework in Section 7, and provide an extension of our algorithms to junction graphs in Section 8. Finally, we present numerical results in Section 9 and conclude the paper in Section 10.

2. Background

We give an overview of different inference problems on graphical models, and introduce the variational framework as applied to max- and sum- inference problems.

2.1 Graphical Models

Let $x = \{x_1, x_2, \cdots, x_n\}$ be a random vector in a discrete space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$. Let $V = \{1, \cdots, n\}$. For an index set $\alpha \subseteq V$, denote by x_α the sub-vector $\{x_i : i \in \alpha\}$, and similarly, $\mathcal{X}_\alpha$ the cross product of $\{\mathcal{X}_i : i \in \alpha\}$. A graphical model defines a factorized probability on x,

$$p(x) = \frac{1}{Z(\psi)} \prod_{\alpha \in I} \psi_\alpha(x_\alpha) \qquad \text{or} \qquad p(x;\theta) = \exp[\sum_{\alpha \in I} \theta_\alpha(x_\alpha) - \Phi(\theta)],$$

where I is a set of subsets of variable indexes, $\psi_\alpha \colon \mathcal{X}_\alpha \to \mathbb{R}^+$ is called a factor function, and $\theta_\alpha(x_\alpha) = \log \psi_\alpha(x_\alpha)$. Since the x_i are discrete, the functions ψ and θ are tables; by alternatively viewing θ as a vector, it is interpreted as the natural parameter in an overcomplete, exponential family representation. Let ψ and θ be the joint vector of all ψ_α and θ_α respectively, for example, $\theta = \{\theta_\alpha(x_\alpha) \colon \alpha \in I, x_\alpha \in \mathcal{X}_\alpha\}$. The normalization constant $Z(\psi)$, called *partition function*, normalizes the probability to sum to one, and $\Phi(\theta) := \log Z(\psi)$ is called the log-partition function,

$$\Phi(\theta) = \log \sum_{x \in \mathcal{X}} \exp[\theta(x)],$$

where we define $\theta(x) = \sum_{\alpha \in I} \theta_\alpha(x_\alpha)$ to be the joint potential function that maps from $\mathcal{X}$ to $\mathbb{R}$. The factorization structure of $p(x)$ can be represented by an undirected graph $G = (V, E)$, where each node $i \in V$ maps to a variable x_i, and each edge $(ij) \in E$ corresponds to two variables x_i and x_j that coappear in some factor function ψ_α, that is, $\{i, j\} \subseteq \alpha$. The set I is then a set of cliques (fully connected subgraphs) of G. For the purpose of illustration, we mainly restrict our scope on the set of pairwise models, on which I is the set of nodes and edges, that is, $I = E \cup V$. However, we show how to extend our algorithms to models with higher order cliques in Section 8.

2.2 Sum-Inference Problems and Variational Approximation

Sum-inference is the task of marginalizing (summing out) variables in the model, for example, calculating the marginal probabilities of single variables, or the normalization constant Z,

$$p(x_i) = \sum_{x_{V \setminus \{i\}}} \exp[\theta(x) - \Phi(\theta)], \qquad \Phi(\theta) = \log \sum_x \exp[\theta(x)].$$

Unfortunately, the problem is generally #P-complete, and the straightforward calculation requires summing over an exponential number of terms. Variational methods are a class of approximation algorithms that transform the marginalization problem into a continuous optimization problem, which is then typically solved approximately.

2.2.1 MARGINAL POLYTOPE

The marginal polytope is a key concept in variational inference. We define the *marginal polytope* $\mathbb{M}$ to be the set of local marginal probabilities $\tau = \{\tau_\alpha(x_\alpha) \colon \alpha \in I\}$ that are extensible to a valid joint distribution, that is,

$$\mathbb{M} = \{\tau \,:\, \exists \text{ joint distribution } q(x), \text{ s.t. } \tau_\alpha(x_\alpha) = \sum_{x_{V \setminus \alpha}} q(x) \text{ for } \forall \alpha \in I\}.$$

Denote by $Q[\tau]$ the set of joint distributions whose marginals are consistent with $\tau \in \mathbb{M}$; by the principle of maximum entropy (Jaynes, 1957), there exists a unique distribution in $Q[\tau]$ that has maximum entropy and follows the exponential family form for some θ.[2] With an abuse of notation, we denote these unique global distributions by $\tau(x)$, and we do not distinguish $\tau(x)$ and τ when it is clear from the context.

2. In the case that $p(x)$ has zero elements, the maximum entropy distribution is still unique and satisfies the exponential family form, but the corresponding θ has negative infinite values (Jaynes, 1957).

2.2.2 LOG-PARTITION FUNCTION DUALITY

A key result to many variational methods is that the log-partition function $\Phi(\theta)$ is a convex function of θ and can be rewritten into a convex dual form,

$$\Phi(\theta) = \max_{\tau \in \mathbb{M}} \big\{ \langle \theta, \tau \rangle + H(\tau) \big\}, \tag{1}$$

where $\langle \theta, \tau \rangle = \sum_\alpha \sum_{x_\alpha} \theta_\alpha(x_\alpha) \tau_\alpha(x_\alpha)$ is the vectorized inner product, and $H(\tau)$ is the entropy of the corresponding global distribution $\tau(x)$, that is, $H(\tau) = -\sum_x \tau(x) \log \tau(x)$. The unique maximum τ^* of (1) exactly equals the marginals of the original distribution $p(x;\theta)$, that is, $\tau^*(x) = p(x;\theta)$. We call $F_{sum}(\tau, \theta) = \langle \theta, \tau \rangle + H(\tau)$ the sum-inference free energy (although technically the *negative* free energy).

The dual form (1) transforms the marginalization problem into a continuous optimization, but does not make it any easier: the marginal polytope $\mathbb{M}$ is defined by an exponential number of linear constraints, and the entropy term in the objective function is as difficult to calculate as the log-partition function. However, (1) provides a framework for deriving efficient approximate inference algorithms by approximating both the marginal polytope and the entropy (Wainwright and Jordan, 2008).

2.2.3 BP-LIKE METHODS

Many approximation methods replace $\mathbb{M}$ with the *locally consistent polytope* $\mathbb{L}$; in pairwise models, it is the set of singleton and pairwise "pseduo-marginals" $\{\tau_i(x_i) \colon i \in V\}$ and $\{\tau_{ij}(x_i, x_j) \colon (ij) \in E\}$ that are consistent on their intersections, that is,

$$\mathbb{L} = \{\tau_i, \tau_{ij} \,:\, \sum_{x_i} \tau_{ij}(x_i, x_j) = \tau_j(x_j),\ \sum_{x_i} \tau_i(x_i) = 1,\ \tau_{ij}(x_i, x_j) \geq 0\}. \tag{2}$$

Since not all such pseudo-marginals have valid global distributions, it is easy to see that $\mathbb{L}$ is an outer bound of $\mathbb{M}$, that is, $\mathbb{M} \subseteq \mathbb{L}$. Note that this means there may not exist a global distribution $\tau(x)$ for τ in $\mathbb{L}$.

The free energy remains intractable (and is not even well-defined) in $\mathbb{L}$. We typically approximate the free energy by a combination of singleton and pairwise entropies, which only requires knowing τ_i and τ_{ij}. For example, the Bethe free energy approximation (Yedidia et al., 2003) is

$$H(\tau) \approx \sum_{i \in V} H_i(\tau) - \sum_{(ij) \in E} I_{ij}(\tau), \qquad \Phi(\theta) \approx \max_{\tau \in \mathbb{L}} \big\{ \langle \theta, \tau \rangle + \sum_{i \in V} H_i(\tau) - \sum_{(ij) \in E} I_{ij}(\tau) \big\}, \tag{3}$$

where $H_i(\tau)$ is the entropy of $\tau_i(x_i)$ and $I_{ij}(\tau)$ the mutual information of x_i and x_j, that is,

$$H_i(\tau) = -\sum_{x_i} \tau_i(x_i) \log \tau_i(x_i), \qquad I_{ij}(\tau) = \sum_{x_i, x_j} \tau_{ij}(x_i, x_j) \log \frac{\tau_{ij}(x_i, x_j)}{\tau_i(x_i) \tau_j(x_j)}.$$

We sometimes abbreviate $H_i(\tau)$ and $I_{ij}(\tau)$ into H_i and I_{ij} for convenience. The well-known loopy belief propagation (BP) algorithm of Pearl (1988) can be interpreted as a fixed point algorithm to optimize the Bethe free energy in (3) on the locally consistent polytope $\mathbb{L}$ (Yedidia et al., 2003). Unfortunately, the Bethe free energy is a non-concave function of τ, causing (3) to be a non-convex

optimization. The tree reweighted (TRW) free energy is a convex surrogate of the Bethe free energy (Wainwright et al., 2005a),

$$\Phi(\theta) \approx \max_{\tau \in \mathbb{L}} \big\{ \langle \theta, \tau \rangle + \sum_{i \in V} H_i(\tau) - \sum_{(ij) \in E} \rho_{ij} I_{ij}(\tau) \big\}, \tag{4}$$

where $\{\rho_{ij} \colon (ij) \in E\}$ is a set of positive edge appearance probabilities obtained from a weighted collection of spanning trees of G (see Wainwright et al. (2005a) and Section 4.2 for the detailed definition). The TRW approximation in (4) is a convex optimization problem, and is guaranteed to give an upper bound of the true log-partition function. A message passing algorithm similar to loopy BP, called tree reweighted BP, can be derived as a fixed point algorithm for solving the convex optimization in (4).

2.2.4 MEAN-FIELD-BASED METHODS

Mean-field-based methods are another set of approximate inference algorithms, which work by restricting $\mathbb{M}$ to a set of tractable distributions, on which both the marginal polytope and the joint entropy are tractable. Precisely, let $\mathbb{M}_{mf}$ be a subset of $\mathbb{M}$ that corresponds to a set of tractable distributions, for example, the set of fully factored distributions, $\mathbb{M}_{mf} = \{\tau \in \mathbb{M} \colon \tau(x) = \prod_{i \in V} \tau_i(x_i)\}$. Note that the joint entropy $H(\tau)$ for any $\tau \in \mathbb{M}_{mf}$ decomposes to the sum of singleton entropies $H_i(\tau)$ of the marginal distributions $\tau_i(x_i)$. This method then approximates the log-partition function (1) by

$$\max_{\tau \in \mathbb{M}_{mf}} \big\{ \langle \theta, \tau \rangle + \sum_{i \in V} H_i(\tau) \big\}, \tag{5}$$

which is guaranteed to give a lower bound of the log-partition function. Unfortunately, mean field methods usually lead to non-convex optimization problems, because $\mathbb{M}_{mf}$ is often a non-convex set. In practice, block coordinate descent methods can be adopted to find the local optima of (5).

2.3 Max-Inference Problems

Combinatorial maximization (max-inference), or maximum *a posteriori* (MAP), problems are the tasks of finding a mode of the joint probability. That is,

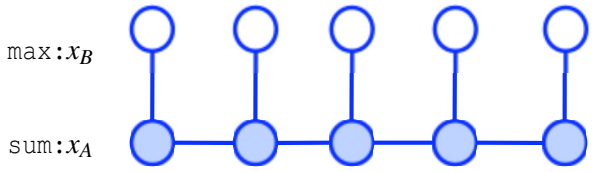
$$\Phi_\infty(\theta) = \max_x \theta(x), \qquad x^* = \arg\max_x \theta(x),$$

where x^* is a MAP configuration and $\Phi_\infty(\theta)$ the optimal energy value. This problem can be reformed into a linear program,

$$\Phi_\infty(\theta) = \max_{\tau \in \mathbb{M}} \langle \theta, \tau \rangle, \tag{6}$$

which attains its maximum when $\tau^*(x) = 1(x = x^*)$, where $1(\cdot)$ is the Kronecker delta function, defined as $1(t) = 1$ if condition t is true, and zero otherwise. If there are multiple MAP solutions, say $\{x^{*k} \colon k = 1, \ldots, K\}$, then any convex combination $\sum_k c_k 1(x = x^{*k})$ with $\sum_k c_k = 1, c_i \geq 0$ leads to a maximum of (6).

The problem in (6) remains NP-hard, because the marginal polytope $\mathbb{M}$ includes exponentially many inequality constraints. Most variational methods for MAP (e.g., Wainwright et al., 2005b; Werner, 2007) can be interpreted as relaxing $\mathbb{M}$ to the locally consistent polytop $\mathbb{L}$, yielding a linear relaxation of the original integer programming problem. Note that (6) differs from (1) only by its lack of an entropy term; in the next section, we generalize this similarity to marginal MAP.



Marginal MAP:

$$x_B^* = \arg\max_{x_B} p(x_B) = \arg\max_{x_B} \sum_{x_A} p(x).$$

Figure 1: An example from Koller and Friedman (2009) in which a marginal MAP query on a tree requires exponential time complexity. The marginalization over x_A destroys the conditional dependency structure in the marginal distribution $p(x_B)$, causing an intractable maximization problem over x_B. The exact variable elimination method, which sequentially marginalizes the sum nodes and then maximizes the max nodes, has time complexity of $O(\exp(n))$, where n is the length of the chain.

2.4 Marginal MAP Problems

Marginal MAP is simply a hybrid of the max- and sum- inference tasks. Let A be a subset of nodes V, and $B = V \backslash A$ be the complement of A. The marginal MAP problem seeks a partial configuration x_B^* that has the maximum marginal probability $p(x_B) = \sum_{x_A} p(x)$, where A is the set of sum nodes to be marginalized out, and B the max nodes to be optimized. We call this a type of "mixed-inference" problem, since it involves more than one type of variable elimination operator. To facilitate developing our duality results, we formulate marginal MAP in terms of the exponential family representation,

$$\Phi_{AB}(\theta) = \max_{x_B} Q(x_B;\theta), \qquad \text{where} \quad Q(x_B;\theta) = \log \sum_{x_A} \exp[\theta(x)], \tag{7}$$

where the maximum point x_B^* of $Q(x_B;\theta)$ is the marginal MAP solution. Although similar to max- and sum-inference, marginal MAP is significantly harder than either of them. A classic example is shown in Figure 1, where marginal MAP is NP-hard even on a tree structured graph (Koller and Friedman, 2009). The main difficulty arises because the max and sum operators do not commute, which restricts feasible elimination orders to those with *all* the sum nodes eliminated before *any* max nodes. In the worst case, marginalizing the sum nodes x_A may destroy any conditional independence among the max nodes x_B, making it difficult to represent or optimize $Q(x_B;\theta)$, even when the sum part alone is tractable (such as when the nodes in A form a tree).

Despite its computational difficulty, marginal MAP plays an essential role in many practical scenarios. The marginal MAP configuration x_B^* in (7) is Bayes optimal in the sense that it minimizes the expected error on B, $\mathbb{E}[1(x_B^* = x_B)]$, where $\mathbb{E}[\cdot]$ denotes the expectation under distribution $p(x;\theta)$. Here, the variables x_A are not included in the error criterion, for example because they are "nuisance" hidden variables of no direct interest, or unobserved or inaccurately measured model parameters. In contrast, the joint MAP configuration x^* minimizes the joint error $\mathbb{E}[1(x^* = x)]$, but gives no guarantees on the partial error $\mathbb{E}[1(x_B^* = x_B)]$. In practice, perhaps because of the wide availability of efficient algorithms for joint MAP, researchers tend to over-use joint MAP even in cases where marginal MAP would be more appropriate. The following toy example shows that this seemingly reasonable approach can sometimes cause serious problems.

Example 1 (Weather Dilemma)**.** *Denote by* $x_b \in \{\texttt{rainy}, \texttt{sunny}\}$ *the weather condition of Irvine, and* $x_a \in \{\texttt{walk}, \texttt{drive}\}$ *whether Alice drives or walks to the school depending on the weather con-*

dition. Assume the probabilities of x_b and x_a are

$p(x_b)$:

rainy	0.4
sunny	0.6

$p(x_a|x_b)$:

	walk	drive
rainy	1/8	7/8
sunny	1/2	1/2

The task is to calculate the most likely weather condition of Irvine, which is obviously sunny *according to $p(x_b)$. The marginal MAP, $x_b^* = \arg\max_{x_b} p(x_b) =$ sunny, gives the correct answer. However, the full MAP estimator, $[x_a^*, x_b^*] = \arg\max p(x_a, x_b) =$ [drive, rainy], gives answer $x_b^* =$ rainy (by dropping the x_a^* component), which is obviously wrong. Paradoxically, if $p(x_a|x_b)$ is changed (say, corresponding to a different person), the solution returned by full MAP could be different.*

In the above example, since no evidence on x_a is observed, the conditional probability $p(x_a|x_b)$ does not provide useful information for x_b, but instead provides misleading information when it is incorporated in the full MAP estimator. The marginal MAP, on the other hand, eliminates the influence of the irrelevant $p(x_a|x_b)$ by marginalizing (or averaging) x_a. In general, the marginal MAP and full MAP can differ significantly when the uncertainty in the hidden variables changes as a function of x_B.

3. A Dual Representation for Marginal MAP

In this section, we present our main result, a dual representation of the marginal MAP problem (7). Our dual representation generalizes that of sum-inference in (1) and max-inference in (6), and provides a unified framework for solving marginal MAP problems.

Theorem 2. *The marginal MAP energy $\Phi_{AB}(\theta)$ in (7) has a dual representation,*

$$\Phi_{AB}(\theta) = \max_{\tau \in \mathbb{M}} \{\langle \theta, \tau \rangle + H_{A|B}(\tau)\}, \tag{8}$$

where $H_{A|B}(\tau)$ is a conditional entropy, $H_{A|B}(\tau) = -\sum_x \tau(x) \log \tau(x_A|x_B)$. If $Q(x_B;\theta)$ has a unique maximum x_B^*, the maximum point τ^* of (8) is also unique, satisfying $\tau^*(x) = \tau^*(x_B)\tau^*(x_A|x_B)$, where $\tau^*(x_B) = 1(x_B = x_B^*)$ and $\tau^*(x_A|x_B) = p(x_A|x_B;\theta)$[3].

Proof. For any $\tau \in \mathbb{M}$ and its corresponding global distribution $\tau(x)$, consider the conditional KL divergence between $\tau(x_A|x_B)$ and $p(x_A|x_B;\theta)$,

$$\begin{aligned} D_{\mathrm{KL}}[\tau(x_A|x_B)||p(x_A|x_B;\theta)] &= \sum_x \tau(x) \log \frac{\tau(x_A|x_B)}{p(x_A|x_B;\theta)} \\ &= -H_{A|B}(\tau) - \mathbb{E}_\tau[\log p(x_A|x_B;\theta)] \\ &= -H_{A|B}(\tau) - \mathbb{E}_\tau[\theta(x)] + \mathbb{E}_\tau[Q(x_B;\theta)] \quad \geq \quad 0, \end{aligned}$$

where $H_{A|B}(\tau)$ is the conditional entropy on $\tau(x)$; the equality on the last line holds because $p(x_A|x_B;\theta) = \exp(\theta(x) - Q(x_B;\theta))$; the last inequality follows from the nonnegativity of KL divergence, and is tight if and only if $\tau(x_A|x_B) = p(x_A|x_B;\theta)$ for all x_A and x_B that $\tau(x_B) \neq 0$. Therefore, we have for any $\tau(x)$,

$$\Phi_{AB}(\theta) = \max_{x_B} Q(x_B;\theta) \geq \mathbb{E}_\tau[Q(x_B;\theta)] \geq \mathbb{E}_\tau[\theta(x)] + H_{A|B}(\tau).$$

3. Since $\tau(x_B) = 0$ if $x_B \neq x_B^*$, we do not necessarily need to define $\tau^*(x_A|x_B)$ for $x_B \neq x_B^*$.

Problem Type	Primal Form	Dual Form
Max-Inference	$\log \max_x \exp(\theta(x))$	$\max_{\tau \in \mathbb{M}} \{\langle \theta, \tau \rangle\}$
Sum-Inference	$\log \sum_x \exp(\theta(x))$	$\max_{\tau \in \mathbb{M}} \{\langle \theta, \tau \rangle + H(\tau)\}$
Marginal MAP	$\log \max_{x_B} \sum_{x_A} \exp(\theta(x))$	$\max_{\tau \in \mathbb{M}} \{\langle \theta, \tau \rangle + H_{A\|B}(\tau)\}$

Table 1: The primal and dual forms of the three inference types. The dual forms of sum-inference and max-inference are well known; the form for marginal MAP is a contribution of this work. Intuitively, the max vs. sum operators in the primal form determine the conditioning set of the conditional entropy term in the dual form.

It is easy to show that the two inequality signs are tight if and only if $\tau(x)$ equals $\tau^*(x)$ as defined above. Substituting $\mathbb{E}_\tau[\theta(x)] = \langle \theta, \tau \rangle$ completes the proof. □

Remark 1. If $Q(x_B; \theta)$ has multiple maxima $\{x_B^{*k}\}$, each corresponding to a distribution $\tau^{*k}(x) = 1(x_B = x_B^*) p(x_A | x_B; \theta)$, then the set of maximum points of (8) is the convex hull of $\{\tau^{*k}\}$.

Remark 2. Theorem 2 naturally integrates the marginalization and maximization sub-problems into one joint optimization problem, providing a novel and efficient treatment for marginal MAP beyond the traditional approaches that treat the marginalization sub-problem as a sub-routine of the maximization problem. As we show in Section 5, this enables us to derive efficient "mixed-product" message passing algorithms that simultaneously takes marginalization and maximization steps, avoiding expensive and possibly wasteful inner loop steps in the marginalization sub-routine.

Remark 3. Since we have $H_{A|B}(\tau) = H(\tau) - H_B(\tau)$ by the entropic chain rule (Cover and Thomas, 2006), the objective function in (8) can be view as a "truncated" free energy,

$$F_{mix}(\tau, \theta) := \langle \theta, \tau \rangle + H_{A|B}(\tau) = F_{sum}(\tau, \theta) - H_B(\tau),$$

where the entropy $H_B(\tau)$ of the max nodes x_B are removed from the regular sum-inference free energy $F_{sum}(\tau, \theta) = \langle \theta, \tau \rangle + H(\tau)$. Theorem 2 generalizes the dual form of both sum-inference (1) and max-inference (6), since it reduces to those forms when the max set B is empty or all nodes, respectively. Table 1 shows all three forms together for comparision. Intuitively, since the entropy $H_B(\tau)$ is removed from the objective, the optimal marginal $\tau^*(x_B)$ tends to have lower entropy and its probability mass concentrates on the optimal configurations $\{x_B^*\}$. Alternatively, the $\tau^*(x)$ can be interpreted as the marginals obtained by clamping the value of x_B at x_B^* on the distribution $p(x; \theta)$, that is, $\tau^*(x) = p(x | x_B = x_B^*; \theta)$.

Remark 4. Unfortunately, subtracting the $H_B(\tau)$ term causes some subtle difficulties. First, $H_B(\tau)$ (and hence $F_{mix}(\tau, \theta)$) may be intractable to calculate even when the joint entropy $H(\tau)$ is tractable, because the marginal distribution $p(x_B) = \sum_{x_A} p(x)$ does not necessarily inherit the conditional dependency structure of the joint distribution. Therefore, the dual optimization in (8) may be intractable even on a tree, reflecting the intrinsic difficulty of marginal MAP compared to full MAP or marginalization. Interestingly, we show in the sequel that a certificate of optimality can still be obtained on general tree graphs in some cases.

Secondly, the conditional entropy $H_{A|B}(\tau)$ (and hence $F_{mix}(\tau,\theta)$) is concave, but not strictly concave, with respect to τ. This creates additional difficulty when optimizing (8), since many iterative optimization algorithms, such as coordinate descent, can lose their typical convergence or optimality guarantees when the objective function is not strongly convex.

3.1 Smoothed Approximation

To sidestep the issue of non-strictly convexity, we introduce a smoothed approximation of $F_{mix}(\tau,\theta)$ that "adds back" part of the missing $H_B(\tau)$ term,

$$F^{\varepsilon}_{mix}(\tau,\theta) = \langle\theta,\tau\rangle + H_{A|B}(\tau) + \varepsilon H_B(\tau),$$

where ε is a small positive constant. Similar smoothing techniques have also been applied to solve the standard MAP problem; see, for example, Hazan and Shashua (2010); Meshi et al. (2012). We show in the following theorem that this smoothed dual approximation is closely connected to a direct approximation in the primal domain.

Theorem 3. *Let ε be a positive constant, and $Q(x_B;\theta)$ as defined in* (7)*. Define*

$$\Phi^{\varepsilon}_{AB}(\theta) = \log\big\{[\sum_{x_B}\exp(Q(x_B;\theta))^{1/\varepsilon}]^{\varepsilon}\big\},$$

then we have

$$\Phi^{\varepsilon}_{AB}(\theta) = \max_{\tau\in\mathbb{M}}\big\{\langle\theta,\tau\rangle + H_{A|B}(\tau) + \varepsilon H_B(\tau)\big\}.$$

In addition, we have

$$\lim_{\varepsilon\to 0^+}\Phi^{\varepsilon}_{AB}(\theta) = \Phi_{AB}(\theta),$$

where $\varepsilon\to 0^+$ denotes approaching zero from the positive side.

Proof. The proof is similar to that of Theorem 2, but exploits the non-negativity of a weighted sum of two KL divergence terms,

$$\mathrm{D}_{KL}[\tau(x_A|x_B)||p(x_A|x_B;\theta)] + \varepsilon\mathrm{D}_{KL}[\tau(x_B)||p(x_B)].$$

The remaining part follows directly from the standard zero temperature limit formula,

$$\lim_{\varepsilon\to 0^+}[\sum_x f(x)^{1/\varepsilon}]^{\varepsilon} = \max_x f(x), \tag{9}$$

where $f(x)$ is any function with positive values. □

4. Variational Approximations for Marginal MAP

Theorem 2 transforms the marginal MAP problem into a variational form, but obviously does not decrease its computational hardness. Fortunately, many well-established variational techniques for sum- and max-inference can be extended to apply to (8), opening a new door for deriving novel approximate algorithms for marginal MAP. In the spirit of Wainwright and Jordan (2008), one can either relax $\mathbb{M}$ to a simpler outer bound like $\mathbb{L}$ and replace $F_{mix}(\tau,\theta)$ by some tractable form to give

algorithms similar to loopy BP or TRW BP, or restrict $\mathbb{M}$ to a tractable subset like $\mathbb{M}_{mf}$ to give mean-field-like algorithms. In the sequel, we demonstrate several such approximation schemes, mainly focusing on the BP-like methods with pairwise free energies. We will briefly discuss mean-field-like methods when we connect to EM in section 7, and derive an extension to junction graphs that exploits higher order approximations in Section 8. Our framework can be easily adopted to take advantage of other, more advanced variational techniques, like those using higher order cliques (e.g., Yedidia et al., 2005; Globerson and Jaakkola, 2007; Liu and Ihler, 2011a; Hazan et al., 2012) or more advanced optimization methods like dual decomposition (Sontag et al., 2011) or alternating direction method of multipliers (Boyd et al., 2010).

We start by characterizing the graph structure on which marginal MAP is tractable.

Definition 4.1. *We call G an A-B* tree *if there exists a partial order on the node set $V = A \cup B$, satisfying*

1) Tree-order. *For any $i \in V$, there is at most one other node $j \in V$ (called its parent), such that $j \prec i$ and $(ij) \in E$;*

2) A-B Consistency. *For any $a \in A$ and $b \in B$, we have $b \prec a$.*

We call such a partial order an A-B tree-order of G.

For further notation, let $G_A = (A, E_A)$ be the subgraph induced by nodes in A, that is, $E_A = \{(ij) \in E : i \in A, j \in A\}$, and similarly for $G_B = (B, E_B)$. Let $\partial_{AB} = \{(ij) \in E : i \in A, j \in B\}$ be the edges that join sets A and B.

Obviously, marginal MAP on an A-B tree can be tractably solved by sequentially eliminating the variables along the A-B tree-order (see, e.g., Koller and Friedman, 2009). We show that its dual optimization is also tractable in this case.

Lemma 4. *If G is an A-B tree, then*

1) *The locally consistent polytope equals the marginal polytope, that is, $\mathbb{M} = \mathbb{L}$.*

2) *The conditional entropy has a pairwise decomposition,*

$$H_{A|B}(\tau) = \sum_{i \in A} H_i(\tau) \; - \sum_{(ij) \in E_A \cup \partial_{AB}} I_{ij}(\tau). \tag{10}$$

Proof. 1) The fact that $\mathbb{M} = \mathbb{L}$ on trees is a standard result; see Wainwright and Jordan (2008) for details.
2) Because G is an A-B tree, both $p(x)$ and $p(x_B)$ have tree structured conditional dependency. We then have (see, e.g., Wainwright and Jordan, 2008) that

$$H(\tau) = \sum_{i \in V} H_i(\tau) - \sum_{(ij) \in E} I_{ij}(\tau), \qquad \text{and} \qquad H_B(\tau) = \sum_{i \in B} H_i(\tau) - \sum_{(ij) \in E_B} I_{ij}(\tau).$$

Equation (10) follows by using the entropic chain rule $H_{A|B}(\tau) = H(\tau) - H_B(\tau)$. □

4.1 Bethe-like Free Energy

Lemma 4 suggests that the free energy of A-B trees can be decomposed into singleton and pairwise terms that are easy to deal with. This is not true for general graphs, but motivates a "Bethe" like approximation,

$$\Phi_{bethe}(\theta) = \max_{\tau \in \mathbb{L}} F_{bethe}(\tau, \theta), \qquad F_{bethe}(\tau, \theta) = \langle \theta, \tau \rangle + \sum_{i \in A} H_i(\tau) - \sum_{(ij) \in E_A \cup \partial_{AB}} I_{ij}(\tau), \tag{11}$$

where $F_{bethe}(\tau, \theta)$ is a "truncated" Bethe free energy, whose entropy and mutual information terms that involve only max nodes are truncated. If G is an A-B tree, Φ_{bethe} equals the true Φ_{AB}, giving an intuitive justification. In the sequel we give more general theoretical conditions under which this approximation gives the exact solution, and we find empirically that it usually gives surprisingly good solutions in practice. Similar to the regular Bethe approximation, (11) leads to a nonconvex optimization, and we will derive both message passing algorithms and provably convergent algorithms to solve it.

4.2 Tree-reweighted Free Energy

Following the idea of TRW belief propagation (Wainwright et al., 2005a), we construct an approximation of marginal MAP using a convex combination of A-B subtrees (subgraphs of G that are A-B trees). Let $\mathcal{T}_{AB}$ be a collection of A-B subtrees of G. We assign with each $T \in \mathcal{T}_{AB}$ a weight w_T satisfying $w_T \geq 0$ and $\sum_{T \in \mathcal{T}_{AB}} w_T = 1$. For each A-B sub-tree $T = (V, E_T)$, define

$$H_{A|B}(\tau \,;\, T) = \sum_{i \in A} H_i(\tau) - \sum_{(ij) \in E_T \setminus E_B} I_{ij}(\tau).$$

As shown in Wainwright and Jordan (2008), the $H_{A|B}(\tau \,;\, T)$ is always a concave function of τ on $\mathbb{L}$, and $H_{A|B}(\tau) \leq H_{A|B}(\tau \,;\, T)$ for all $\tau \in \mathbb{M}$ and $T \in \mathcal{T}_{AB}$. More generally, we have $H_{A|B}(\tau) \leq \sum_{T \in \mathcal{T}_{AB}} w_T H_{A|B}(\tau \,;\, T)$, which can be transformed to

$$H_{A|B}(\tau) \leq \sum_{i \in A} H_i(\tau) - \sum_{(ij) \in E_A \cup \partial_{AB}} \rho_{ij} I_{ij}(\tau), \tag{12}$$

where $\rho_{ij} = \sum_{T : (ij) \in E_T} w_T$ are the edge appearance probabilities as defined in Wainwright and Jordan (2008). Replacing $\mathbb{M}$ with $\mathbb{L}$ and $H_{A|B}(\tau)$ with the bound in (12) leads to a TRW-like approximation of marginal MAP,

$$\Phi_{trw}(\theta) = \max_{\tau \in \mathbb{L}} F_{trw}(\tau, \theta), \qquad F_{trw}(\tau, \theta) = \langle \theta, \tau \rangle + \sum_{i \in A} H_i(\tau) - \sum_{(ij) \in E_A \cup \partial_{AB}} \rho_{ij} I_{ij}(\tau). \tag{13}$$

Since $\mathbb{L}$ is an outer bound of $\mathbb{M}$, and F_{trw} is a concave upper bound of the true free energy, we can guarantee that $\Phi_{trw}(\theta)$ is always an upper bound of $\Phi_{AB}(\theta)$. To our knowledge, this provides the first known convex relaxation for upper bounding marginal MAP. One can also optimize the weights $\{w_T : T \in \mathcal{T}_{AB}\}$ to get the tightest upper bound using methods similar to those used for regular TRW BP (see Wainwright et al., 2005a).

4.3 Global Optimality Guarantees

We show the global optimality guarantees of the above approximations under some circumstances. In this section, we always assume G_A is a tree, and hence the objective function is tractable to

calculate for a given x_B. However, the optimization component remains intractable in this case, because the marginalization step destroys the decomposition structure of the objective function (see Figure 1). It is thus nontrivial to see how the Bethe and TRW approximations behave in this case.

In general, suppose we approximate $\Phi_{AB}(\theta)$ using the following pairwise approximation,

$$\Phi_{tree}(\theta) = \max_{\tau \in \mathbb{L}} \Big\{ \langle \theta, \tau \rangle \; + \sum_{i \in A} H_i(\tau) - \sum_{(ij) \in E_A} I_{ij}(\tau) - \sum_{(ij) \in \partial_{AB}} \rho_{ij} I_{ij}(\tau) \Big\}, \tag{14}$$

where the weights on the sum part, $\{\rho_{ij} \colon (ij) \in E_A\}$, have been fixed to be ones. This choice makes sure that the sum part is "intact" in the approximation, while the weights on the crossing edges, $\rho_{AB} = \{\rho_{ij} \colon (ij) \in \partial_{AB}\}$, can take arbitrary values, corresponding to different free energy approximation methods. If $\rho_{ij} = 1$ for $\forall (ij) \in \partial_{AB}$, it is the Bethe free energy; it will correspond to the TRW free energy if $\{\rho_{ij}\}$ are taken to be a set of edge appearance probabilities (which in general have values less than one). The edge appearance probabilities of *A*-*B* trees are more restrictive than for the standard trees used in TRW BP. For example, if the max part of a *A*-*B* sub-tree is a connected tree, then it can include at most one crossing edge, so in this case ρ_{AB} should satisfy $\sum_{(ij) \in \partial_{AB}} \rho_{ij} = 1$, $\rho_{ij} \geq 0$. Interestingly, we will show in Section 7 that if $\rho_{ij} \to +\infty$ for $\forall (ij) \in \partial_{AB}$, then Equation (14) is closely related to an EM algorithm.

Theorem 5. *Suppose the sum part G_A is a tree, and we approximate $\Phi_{AB}(\theta)$ using $\Phi_{tree}(\theta)$ defined in* (14)*. Assume that* (14) *is* globally *optimized.*

(i) *We have $\Phi_{tree}(\theta) \geq \Phi_{AB}(\theta)$. If there exists x_B^* such that $Q(x_B^*; \theta) = \Phi_{tree}(\theta)$, we have $\Phi_{tree}(\theta) = \Phi_{AB}(\theta)$, and x_B^* is a globally optimal marginal MAP solution.*

(ii) *Suppose τ^* is a* global *maximum of* (14)*, and $\{\tau_i^*(x_i) \colon i \in B\}$ have integral values, that is, $\tau_i^*(x_i) = 0$ or 1, then $\{x_i^* = \arg\max_{x_i} \tau_i^*(x_i) \colon i \in B\}$ is a globally optimal solution of the marginal MAP problem* (7)*.*

Proof (sketch). (See appendix for the complete proof.) The fact that the sum part G_A is a tree guarantees the marginalization is exact. Showing (14) is a relaxation of the maximization problem and applying standard relaxation arguments completes the proof. □

Remark. Theorem 5 works for arbitrary values of ρ_{AB}, and suggests a fundamental tradeoff of hardness as ρ_{AB} takes on different values. On the one hand, the value of ρ_{AB} controls the concavity of the objective function in (14) and hence the difficulty of finding a global optimum; small enough ρ_{AB} (as in TRW) can ensure that (14) is a convex optimization, while larger ρ_{AB} (as in Bethe or EM) causes (14) to become non-convex, making it difficult to apply Thoerem 5. On the other hand, the value of ρ_{AB} also controls how likely the solution is to be integral—larger ρ_{ij} emphasizes the mutual information terms, forcing the solution towards integral points. Thus the solution of the TRW free energy is less likely to be integral than the Bethe free energy, causing a difficulty in applying Theorem 5 to TRW solutions as well. The TRW approximation ($\sum_{ij} \rho_{ij} = 1$) and EM ($\rho_{ij} \to +\infty$; see Section 7) reflect two extrema of this tradeoff between concavity and integrality, respectively, while the Bethe approximation ($\rho_{ij} = 1$) appears to represent a reasonable compromise that often gives excellent performance in practice. In Section 5.2, we give a different set of local optimality guarantees that are derived from a reparameterization perspective.

5. Message Passing Algorithms for Marginal MAP

We now derive message-passing-style algorithms to optimize the "truncated" Bethe or TRW free energies in (11) and (13). Instead of optimizing the truncated free energies directly, we leverage the results of Theorem 3 and consider their "annealed" versions,

$$\max_{\tau\in\mathbb{L}} \big\{ \langle \theta, \tau\rangle + \hat{H}_{A|B}(\tau) + \varepsilon \hat{H}_B(\tau) \big\},$$

where ε is a positive annealing coefficient (or temperature), and the $\hat{H}_{A|B}(\tau)$ and $\hat{H}_B(\tau)$ are the generic pairwise approximations of $H_{A|B}(\tau)$ and $H_B(\tau)$, respectively. That is,

$$\hat{H}_{A|B}(\tau) = \sum_{i\in A} H_i(\tau) \; - \sum_{(ij)\in E_A\cup\partial_{AB}} \rho_{ij} I_{ij}(\tau), \qquad \text{and} \qquad \hat{H}_B(\tau) = \sum_{i\in B} H_i(\tau) \; - \sum_{(ij)\in E_B} \rho_{ij} I_{ij}(\tau), \tag{15}$$

where different values of pairwise weights $\{\rho_{ij}\}$ correspond to either the Bethe approximation or the TRW approximation. This yields a generic pairwise free energy optimization problem,

$$\max_{\tau\in\mathbb{L}} \big\{ \langle \theta, \tau\rangle + \sum_{i\in V} w_i H_i(\tau) - \sum_{(ij)\in E} w_{ij} I_{ij}(\tau) \big\}, \tag{16}$$

where the weights $\{w_i, w_{ij}\}$ are determined by the temperature ε and $\{\rho_{ij}\}$ via

$$w_i = \begin{cases} 1 & \forall i \in A \\ \varepsilon & \forall i \in B, \end{cases} \qquad\qquad w_{ij} = \begin{cases} \rho_{ij} & \forall (ij) \in E_A \cup \partial_{AB} \\ \varepsilon\rho_{ij} & \forall (ij) \in E_B. \end{cases} \tag{17}$$

The general framework in (16) provides a unified treatment for approximating sum-inference, max-inference and mixed, marginal MAP problems simply by taking different weights. Specifically,

1. If $w_i = 1$ for all $i \in V$, Equation (16) corresponds to the sum-inference problem and the sum-product BP objectives and algorithms.

2. If $w_i \to 0^+$ for all $i \in V$ (and the corresponding $w_{ij} \to 0^+$), Equation (16) corresponds to the max-inference problem and the max-product linear programming objective and algorithms.

3. If $w_i = 1$ for $\forall i \in A$ and $w_i = 0$ for $\forall i \in B$ (and the corresponding $w_{ij} \to 0^+$), Equation (16) corresponds to the marginal MAP problem; in the sequel, we derive "mixed-product" BP algorithms.

Note the different roles of the singleton and pairwise weights: the singleton weights $\{w_i \colon i \in V\}$ define the type of inference problem, while the pairwise weights $\{w_{ij} \colon (ij) \in E\}$ determine the approximation method (e.g., Bethe vs. TRW).

We now derive a message passing algorithm for solving the generic problem (16), using a Lagrange multiplier method similar to Yedidia et al. (2005) or Wainwright et al. (2005a).

Proposition 6. *Assuming w_i and w_{ij} are strictly positive, the stationary points of* (16) *satisfy the fixed point condition of the following message passing update,*

Message Update:
$$m_{i\to j}(x_j) \leftarrow \Big[\sum_{x_i} \big(\psi_i(x_i) m_{\sim i}(x_i)\big)^{\frac{1}{w_i}} \left(\frac{\psi_{ij}(x_i,x_j)}{m_{j\to i}(x_i)} \right)^{\frac{1}{w_{ij}}} \Big]^{w_{ij}}, \tag{18}$$

Marginal Decoding:

$$\tau_i(x_i) \propto \big[\psi_i(x_i) m_{\sim i}(x_i)\big]^{\frac{1}{w_i}}, \quad \tau_{ij}(x_i,x_j) \propto \tau_i(x_i)\tau_j(x_j) \left[\frac{\psi_{ij}(x_i,x_j)}{m_{i\to j}(x_j) m_{j\to i}(x_i)} \right]^{\frac{1}{w_{ij}}}, \tag{19}$$

Algorithm 1 Annealed BP for Marginal MAP

Define the pairwise weights $\{\rho_{ij}\colon (ij) \in E\}$, for example, $\rho_{ij} = 1$ for Bethe or valid appearance probabilities for TRW. Initialize the messages $\{m_{i\to j}\colon (ij) \in E\}$.

for iteration t **do**

1. Update ε by $\varepsilon = 1/t$, and correspondingly the weights $\{w_i, w_{ij}\}$ by (17).
2. Perform the message passing update in (18) for all edges $(ij) \in E$.

end for

Calculate the singleton beliefs $b_i(x_i)$ and decode the solution x_B^*,

$$x_i^* = \arg\max_{x_i} b_i(x_i), \quad \forall i \in B, \text{ where } b_i(x_i) \propto \psi_i(x_i) m_{\sim i}(x_i).$$

where $m_{\sim i}(x_i) := \prod_{k \in \partial_i} m_{k\to i}(x_i)$ *is the product of messages sent into node i, and* ∂_i *is the set of neighboring nodes of i.*

Proof (sketch). (See appendix for the complete proof.) Note that (19) is simply the KKT condition of (16), with the log of the message $\log m_{i\to j}$ being the Lagrange multipliers. Plugging (19) into the local consistency constraints of $\mathbb{L}$ in (2) gives (18). □

The above message update is mostly similar to TRW-BP of Wainwright et al. (2005a), except that it incorporates general singleton weights w_i. The marginal MAP problem can be solved by running (18) with $\{w_i, w_{ij}\}$ defined by (17) and a scheme for choosing the temperature ε, either directly set to be a small constant, or gradually decreased (or annealed) to zero through iterations, for example, by $\varepsilon = 1/t$ where t is the iteration. Algorithm 1 describes the details for the annealing method.

5.1 Mixed-Product Belief Propagation

Directly taking $\varepsilon \to 0^+$ in message update (18), we can get an interesting "mixed-product" BP algorithm that is a hybrid of the max-product and sum-product message updates, with a novel "argmax-product" message update that is specific to marginal MAP problems. This algorithm is listed in Algorithm 2, and described by the following proposition:

Proposition 7. *As* ε *approaches zero from the positive side, that is,* $\varepsilon \to 0^+$*, the message update* (18) *reduces to the update in* (20)-(22) *in Algorithm 2.*

Proof. For messages from $i \in A$ to $j \in A \cup B$, we have $w_i = 1$, $w_{ij} = \rho_{ij}$; the result is obvious.
For messages from $i \in B$ to $j \in B$, we have $w_i = \varepsilon$, $w_{ij} = \varepsilon\rho_{ij}$. The result follows from the zero temperature limit formula in (9), by letting $f(x_i) = (\psi_i(x_i) m_{\sim i}(x_i))^{\rho_{ij}} (\frac{\psi_{ij}(x_i,x_j)}{m_{j\to i}(x_i)})$.
For messages from $i \in B$ to $j \in A$, we have $w_i = \varepsilon$, $w_{ij} = \rho_{ij}$. One can show that

$$\lim_{\varepsilon \to 0^+} \left[\frac{\psi_i(x_i) m_{\sim i}(x_i)}{\max_{x_i} \psi_i(x_i) m_{\sim i}(x_i)}\right]^{1/\varepsilon} = 1(x_i \in \mathcal{X}_i^*),$$

where $\mathcal{X}_i^* = \arg\max_{x_i} \psi_i(x_i) m_{\sim i}(x_i)$. Plugging this into (18) and dropping the constant term, we get the message update in (22). □

Algorithm 2 Mixed-product Belief Propagation for Marginal MAP

Define the pairwise weights $\{\rho_{ij}\colon (ij) \in E\}$ and initialize messages $\{m_{i\to j}\colon (ij) \in E\}$ as in Algorithm 1.

for iteration t **do**

for edge $(ij) \in E$ **do**

Perform different message updates depending on the node type of the source and destination,

$$A \to A \cup B\text{:} \quad m_{i\to j}(x_j) \leftarrow \Big[\sum_{x_i}(\psi_i(x_i)m_{\sim i}(x_i))\Big(\frac{\psi_{ij}(x_i,x_j)}{m_{j\to i}(x_i)}\Big)^{1/\rho_{ij}}\Big]^{\rho_{ij}}, \tag{20}$$

$$B \to B\text{:} \quad m_{i\to j}(x_j) \leftarrow \max_{x_i}(\psi_i(x_i)m_{\sim i}(x_i))^{\rho_{ij}}\Big(\frac{\psi_{ij}(x_i,x_j)}{m_{j\to i}(x_i)}\Big), \tag{21}$$

$$B \to A\text{:} \quad m_{i\to j}(x_j) \leftarrow \Big[\sum_{x_i\in\mathcal{X}_i^*}(\psi_i(x_i)m_{\sim i}(x_i))\Big(\frac{\psi_{ij}(x_i,x_j)}{m_{j\to i}(x_i)}\Big)^{1/\rho_{ij}}\Big]^{\rho_{ij}}, \tag{22}$$

(sum-product) / (max-product) / (argmax-product)

where the set $\mathcal{X}_i^* = \arg\max_{x_i} \psi_i(x_i)m_{\sim i}(x_i)$ and $m_{\sim i}(x_i) = \prod_{k\in\partial_i} m_{ki}(x_i)$.

end for

end for

Calculate the singleton beliefs $b_i(x_i)$ and decode the solution x_B^*,

$$x_i^* = \arg\max_{x_i} b_i(x_i), \quad \forall i \in B, \text{ where } b_i(x_i) \propto \psi_i(x_i)m_{\sim i}(x_i).$$

Algorithm 2 has an intuitive interpretation: the sum-product and max-product messages in (20) and (21) correspond to the marginalization and maximization steps, respectively. The special "argmax-product" messages in (22) serves to synchronize the sum-product and max-product messages—it restricts the max nodes to the currently decoded local marginal MAP solutions $\mathcal{X}_i^* = \arg\max \psi_i(x_i)m_{\sim i}(x_i)$, and passes the posterior beliefs back to the sum part. Note that the summation notation in (22) can be ignored if $\mathcal{X}_i^*$ has only a single optimal state.

One critical feature of our mixed-product BP is that it takes simultaneous movements on the marginalization and maximization sub-problems in a parallel fashion, and is computationally much more efficient than the traditional methods that require fully solving a marginalization sub-problem before taking each maximization step. This advantage is inherited from our general variational framework, which naturally integrates the marginalization and maximization sub-problems into a joint optimization problem.

Interestingly, Algorithm 2 also bears similarity to a recent hybrid message passing method of Jiang et al. (2011), which differs from Algorithm 2 only in replacing the special argmax-product messages (22) with regular max-product messages. We make a detailed comparison of these two algorithms in Section 5.3, and show that it is in fact the argmax-product messages (22) that lends our algorithm several appealing optimality guarantees.

5.2 Reparameterization Interpretation and Local Optimality Guarantees

An important interpretation of the sum-product and max-product BP is the reparameterization viewpoint (Wainwright et al., 2003; Weiss et al., 2007): Message passing updates can be viewed as moving probability mass between local pseudo-marginals (or beliefs), in a way that leaves their product a reparameterization of the original distribution, while ensuring some consistency conditions at the fixed points. Such viewpoints are theoretically important, because they are useful for proving optimality guarantees for the BP algorithms. In this section, we show that the mixed-product BP in Algorithm 2 has a similar reparameterization interpretation, based on which we establish a local optimality guarantee for mixed-product BP.

To start, we define a set of "mixed-beliefs" as

$$b_i(x_i) \propto \psi_i(x_i) m_{\sim i}(x_i), \qquad b_{ij}(x_{ij}) \propto b_i(x_i) b_j(x_j) \left[\frac{\psi_{ij}(x_i,x_j)}{m_{i\to j}(x_j) m_{j\to i}(x_i)} \right]^{1/\rho_{ij}}. \tag{23}$$

The marginal MAP solution should be decoded from $x_i^* \in \arg\max_{x_i} b_i(x_i), \forall i \in B$, as is typical in max-product BP. Note that the above mixed-beliefs $\{b_i, b_{ij}\}$ are different from the local marginals $\{\tau_i, \tau_{ij}\}$ defined in (19), but are rather softened versions of $\{\tau_i, \tau_{ij}\}$.Their relationship is explicitly clarified in the following.

Proposition 8. *The* $\{\tau_i, \tau_{ij}\}$ *in* (19) *and the* $\{b_i, b_{ij}\}$ *in* (23) *are associated via,*

$$\begin{cases} b_i \propto \tau_i & \forall i \in A, \\ b_i \propto (\tau_i)^{\varepsilon} & \forall i \in B \end{cases} \qquad \begin{cases} b_{ij} \propto b_i b_j (\frac{\tau_{ij}}{\tau_i \tau_j}) & \forall (ij) \in E_A \cup \partial_{AB} \\ b_{ij} \propto b_i b_j (\frac{\tau_{ij}}{\tau_i \tau_j})^{\varepsilon} & \forall (ij) \in E_B. \end{cases}$$

Proof. Result follows from the simple algebraic transformation between (19) and (23). □

Therefore, as $\varepsilon \to 0^+$, the τ_i $(= b_i^{1/\varepsilon})$ for $i \in B$ should concentrate their mass on a deterministic configuration, but b_i may continue to have soft values.

We now show that the mixed-beliefs $\{b_i, b_{ij}\}$ have a reparameterization interpretation.

Theorem 9. *At the fixed point of mixed-product BP in Algorithm 2 , the mixed-beliefs defined in* (23) *satisfy*
Reparameterization:

$$p(x) \propto \prod_{i \in V} b_i(x_i) \prod_{(ij) \in E} [\frac{b_{ij}(x_i,x_j)}{b_i(x_i) b_j(x_j)}]^{\rho_{ij}}. \tag{24}$$

Mixed-consistency:

$$\text{(a)} \qquad \sum_{x_i} b_{ij}(x_i,x_j) = b_j(x_j), \qquad \forall i \in A, j \in A \cup B, \tag{25}$$

$$\text{(b)} \qquad \max_{x_i} b_{ij}(x_i,x_j) = b_j(x_j), \qquad \forall i \in B, j \in B, \tag{26}$$

$$\text{(c)} \qquad \sum_{x_i \in \arg\max b_i} b_{ij}(x_i,x_j) = b_j(x_j), \qquad \forall i \in B, j \in A. \tag{27}$$

Proof. Directly substitute the definition (23) into the message update (20)-(22). □

The three mixed-consistency constraints exactly map to the three types of message updates in Algorithm 2. Constraint (a) and (b) enforces the regular sum- and max- consistency of the sum- and max- product messages in (20) and (21), respectively. Constraint (c) corresponds to the argmax-product message update in (22): it enforces the marginals to be consistent after x_i is assigned to the currently decoded solution, $x_i = \arg\max_{x_i} b_i(x_i) = \arg\max_{x_i} \sum_{x_j} b_{ij}(x_i, x_j)$, corresponding to solving a local marginal MAP problem on $b_{ij}(x_i, x_j)$. It turns out that this special constraint is a crucial ingredient of mixed-product BP, enabling us to prove guarantees on the strong local optimality of the solution.

Some notation is required. Suppose C is a subset of max nodes in B. Let $G_{C\cup A} = (C \cup A, E_{C\cup A})$ be the subgraph of G induced by nodes $C \cup A$, where $E_{C\cup A} = \{(ij) \in E : i, j \in C \cup A\}$. We call $G_{C\cup A}$ a semi-A-B subtree of G if the edges in $E_{C\cup A} \backslash E_B$ form an A-B tree. In other words, $G_{C\cup A}$ is a semi-A-B tree if it is an A-B tree when ignoring any edges entirely within the max set B. See Figure 2 for examples of semi A-B trees.

Following Weiss et al. (2007), we say that a set of weights $\{\rho_{ij}\}$ is *provably convex* if there exist positive constants κ_i and $\kappa_{i\to j}$, such that $\kappa_i + \sum_{i' \in \partial_i} \kappa_{i'\to i} = 1$ and $\kappa_{i\to j} + \kappa_{j\to i} = \rho_{ij}$. Weiss et al. (2007) shows that if $\{\rho_{ij}\}$ is provably convex, then $H(\tau) = \sum_i H_i(\tau) - \sum_{ij} \rho_{ij} I_{ij}(\tau)$ is a concave function of τ in the locally consistent polytope $\mathbb{L}$.

Theorem 10. *Suppose C is a subset of B such that $G_{C\cup A}$ is a semi-A-B tree, and the weights $\{\rho_{ij}\}$ satisfy*

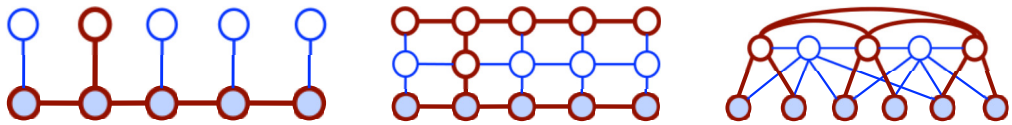
1. $\rho_{ij} = 1$ *for* $(ij) \in E_A$;
2. $0 \le \rho_{ij} \le 1$ *for* $(ij) \in E_{C\cup A} \cap \partial_{AB}$;
3. $\{\rho_{ij} : (ij) \in E_{C\cup A} \cap E_B\}$ *is provably convex.*

At the fixed point of mixed-product BP in Algorithm 2, if the mixed-beliefs on the max nodes $\{b_i, b_{ij} : i, j \in B\}$ defined in (23) *all have unique maxima, then there exists a B-configuration x_B^* satisfying $x_i^* = \arg\max b_i$ for $\forall i \in B$ and $(x_i^*, x_j^*) = \arg\max b_{ij}$ for $\forall (ij) \in E_B$, and x_B^* is locally optimal in the sense that $Q(x_B^*; \theta)$ is not smaller than any B-configuration that differs from x_B^* only on C, that is, $Q(x_B^*; \theta) = \max_{x_C} Q([x_C, x_{B\backslash C}^*]; \theta)$.*

Proof (sketch). (See appendix for the complete proof.) The mixed-consistency constraint (c) in (27) and the fact that $G_{C\cup A}$ is a semi-A-B tree enables the summation part to be eliminated away. The remaining part only involves the max nodes, and the method in Weiss et al. (2007) for analyzing standard MAP can be applied. □

Remark. The proof of Theorem 10 relies on transforming the marginal MAP problem to a standard MAP problem by eliminating the summation part. Therefore, variants of Theorem 10 may be derived using other global optimality conditions of convexified belief propagation or linear programming algorithms for MAP, such as those in Werner (2007, 2010); Wainwright et al. (2005b). We leave this to future work.

For $G_{C\cup A}$ to be a semi A-B tree, the sum part G_A must be a tree, which Theorem 10 assumes implicitly. For the hidden Markov chain in Figure 1, Theorem 10 implies only the local optimality up to Hamming distance one (or coordinate-wise optimality), because any semi A-B subtree of G in Figure 1 can contain at most one max node. However, Theorem 10 is in general much stronger, especially when the sum part is not fully connected, or when the max part has interior regions disconnected from the sum part. As examples, see Figure 2(b)-(c).



(a) (b) (c)

Figure 2: Examples of semi A-B trees. The shaded nodes represent sum nodes, while the unshaded are max nodes. In each graph, a semi A-B tree is labeled by red bold lines. Under the conditions of Theorem 10, the fixed point of mixed-product BP is locally optimal up to jointly perturbing all the max nodes in any semi-A-B subtree of G.

5.3 The Importance of the Argmax-product Message Updates

Jiang et al. (2011) proposed a similar hybrid message passing algorithm, repeated here as Algorithm 3, which differs from our mixed-product BP only in replacing our argmax-product message update (22) with the usual max-product message update (21). We show in this section that this very difference gives Algorithm 3 very different properties, and fewer optimality guarantees, than our mixed-product BP.

Algorithm 3 Hybrid Message Passing by Jiang et al. (2011)

1. Message Update:

$$A \to A \cup B\text{: (sum-product)} \qquad m_{i\to j}(x_j) \leftarrow \Big[\sum_{x_i}(\psi_i(x_i)m_{\sim i}(x_i))\Big(\frac{\psi_{ij}(x_i,x_j)}{m_{j\to i}(x_i)}\Big)^{1/\rho_{ij}}\Big]^{\rho_{ij}},$$

$$A \to A \cup B\text{: (max-product)} \qquad m_{i\to j}(x_j) \leftarrow \max_{x_i}(\psi_i(x_i)m_{\sim i}(x_i))^{\rho_{ij}}\Big(\frac{\psi_{ij}(x_i,x_j)}{m_{j\to i}(x_i)}\Big).$$

2. Decoding: $x_i^* = \arg\max_{x_i} b_i(x_i)$ for $\forall i \in B$, where $b_i(x_i) \propto \psi_i(x_i)m_{\sim i}(x_i)$.

Similar to our mixed-product BP, Algorithm 3 also satisfies the reparameterization property in (24) (with beliefs $\{b_i, b_{ij}\}$ defined by (23)); it also satisfies a set of similar, but crucially different, consistency conditions at its fixed points,

$$\sum_{x_i} b_{ij}(x_i,x_j) = b_j(x_j), \qquad \forall i \in A, j \in A \cup B,$$

$$\max_{x_i} b_{ij}(x_i,x_j) = b_j(x_j), \qquad \forall i \in B, j \in A \cup B,$$

which exactly map to the max- and sum- product message updates in Algorithm 3.

Despite its striking similarity, Algorithm 3 has very different properties, and does not share the appealing variational interpretation and optimality guarantees that we have demonstrated for mixed-product BP. First, it is unclear whether Algorithm 3 can be interpreted as a fixed point algorithm for maximizing our, or a similar, variational objective function. Second, it does not inherit the same optimality guarantees in Theorem 10, despite its similar reparameterization and consistency conditions. These disadvantages are caused by the miss of the special argmax-product message update and its associated mixed-consistency condition in (27), which was a critical ingredient of the proof of Theorem 10.

More detailed insights into Algorithm 3 and mixed-product BP can be obtained by considering the special case when the full graph G is an undirected tree. We show that in this case, Algorithm 3 can be viewed as optimizing a set of *approximate* objective functions, obtained by rearranging the max and sum operators into orders that require less computational cost, while mixed-product BP attempts to maximize the *exact* objective function by message updates that effectively perform some "asynchronous" coordinate descent steps. In the sequel, we use an illustrative toy example to explain the main ideas.

Example 2. Consider a marginal MAP problem on a four node chain-structured graphical model $x_3 - x_1 - x_2 - x_4$, where the sum and max sets are $A = \{1,2\}$ and $B = \{3,4\}$, respectively. We analyze how Algorithm 3 and mixed-product BP in Algorithm 2 perform on this toy example, when both taking Bethe weights ($\rho_{ij} = 1$ for $(ij) \in E$).

Algorithm 3 (Jiang et al. 2011). Since G is a tree, one can show that Algorithm 3 (with Bethe weights) terminates after a full forward and backward iteration (e.g., messages passed along $x_3 \to x_1 \to x_2 \to x_4$ and then $x_4 \to x_2 \to x_1 \to x_3$). By tracking the messages, one can write its final decoded solution in a closed form,

$$x_3^* = \arg\max_{x_3} \sum_{x_1} \sum_{x_2} \max_{x_4} [\exp(\theta(x))], \qquad x_4^* = \arg\max_{x_4} \sum_{x_2} \sum_{x_1} \max_{x_3} [\exp(\theta(x))],$$

On the other hand, the true marginal MAP solution is given by,

$$x_3^* = \arg\max_{x_3} \max_{x_4} \sum_{x_1} \sum_{x_2} [\exp(\theta(x))], \qquad x_4^* = \arg\max_{x_4} \max_{x_3} \sum_{x_2} \sum_{x_1} [\exp(\theta(x))].$$

Here, Algorithm 3 approximates the exact marginal MAP problem by rearranging the max and sum operators into an elimination order that makes the calculation easier. A similar property holds for the general case when G is undirected tree: Algorithm 3 (with Bethe weights) terminates in a finite number of steps, and its output solution x_i^* effectively maximizes an approximate objective function obtained by reordering the max and sum operators along a tree-order (see Definition 4.1) that is rooted at node i. The performance of the algorithm should be related to the error caused by exchanging the order of max and sum operators. However, exact optimality guarantees are likely difficult to show because it maximizes an inexact objective function. In addition, since each component x_i^* uses a different order of arrangement, and hence maximizes a different surrogate objective function, it is unclear whether the joint B-configuration $x_B^* = \{x_i^* : i \in B\}$ given by Algorithm 3 maximizes a single consistent objective function.

Algorithm 2 (mixed-product). On the other hand, the mixed-product belief propagation in Algorithm 2 may not terminate in a finite number of steps, nor does it necessarily yield a closed form solution when G is an undirected tree. However, Algorithm 2 proceeds in an attempt to optimize the exact objective function. In this toy example, we can show that the true solution is guaranteed to be a fixed point of Algorithm 2. Let $b_3(x_3)$ be the mixed-belief on x_3 at the current iteration, and $x_3^* = \arg\max_{x_3} b_3(x_3)$ its unique maxima. After a message sequence passed from x_3 to x_4, one can show that $b_4(x_4)$ and x_4^* update to

$$x_4^* = \arg\max_{x_4} b_4(x_4), \qquad b_4(x_4) = \sum_{x_2} \sum_{x_1} \exp(\theta([x_3^*, x_{\neg 3}])) = \exp(Q([x_3^*, x_4]; \theta)),$$

where we maximize the exact objective function $Q([x_3, x_4]; \theta)$ with fixed $x_3 = x_3^*$. Therefore, on this toy example, one sweep ($x_3 \to x_4$ or $x_4 \to x_3$) of Algorithm 2 is effectively performing a coordinate

Algorithm 4 Proximal Point Algorithm for Marginal MAP (Exact)

Initialize local marginals τ^0.
for iteration t **do**

$$\theta^{t+1} = \theta + \lambda^t \log \tau_B^t, \tag{28}$$

$$\tau^{t+1} = \arg\max_{\tau \in \mathbb{M}} \{\langle \tau, \theta^{t+1} \rangle + H_{A|B}(\tau) + \lambda^t H_B(\tau)\}, \tag{29}$$

end for
Decoding: $x_i^* = \arg\max_{x_i} \tau_i(x_i)$ for $\forall i \in B$.

descent step, which monotonically improves the true objective function towards a local maximum. In more general models, Algorithm 2 differs from sequential coordinate descent, and does not guarantee monotonic convergence. But, it can be viewed as a "parallel" version of coordinate descent, which ensures the stronger local optimality guarantees shown in Theorem 10.

6. Convergent Algorithms by Proximal Point Methods

An obvious disadvantage of mixed-product BP is its lack of convergence guarantees, even when G is an undirected tree. In this section, we apply a proximal point approach (e.g., Martinet, 1970; Rockafellar, 1976) to derive convergent algorithms that directly optimize our free energy objectives, which take the form of transforming marginal MAP into a sequence of pure (or annealed) sum-inference tasks. Similar methods have been applied to standard sum-inference (Yuille, 2002) and max-inference (Ravikumar et al., 2010).

For the purpose of illustration, we first consider the problem of maximizing the *exact* marginal MAP free energy, $F_{mix}(\tau, \theta) = \langle \tau, \theta \rangle + H_{A|B}(\tau)$. The proximal point algorithm works by iteratively optimizing a smoothed problem,

$$\tau^{t+1} = \arg\min_{\tau \in \mathbb{M}} \{-F_{mix}(\tau, \theta) + \lambda^t D(\tau || \tau^t)\},$$

where τ^t is the solution at iteration t, and λ^t is a positive coefficient. Here, $D(\cdot||\cdot)$ is a distance, called the proximal function, which forces τ^{t+1} to be close to τ^t; typical choices of $D(\cdot||\cdot)$ are Euclidean or Bregman distances or ψ-divergences (e.g., Teboulle, 1992; Iusem and Teboulle, 1993). Proximal algorithms have nice convergence guarantees: the objective series $\{f(\tau^t)\}$ is guaranteed to be non-increasing at each iteration, and $\{\tau^t\}$ converges to an optimal solution, under some regularity conditions. See, for example, Rockafellar (1976); Tseng and Bertsekas (1993); Iusem and Teboulle (1993). The proximal algorithm is closely related to the majorize-minimize (MM) algorithm (Hunter and Lange, 2004) and the convex-concave procedure (Yuille, 2002).

For our purpose, we take $D(\cdot||\cdot)$ to be a KL divergence between distributions on the max nodes,

$$D(\tau||\tau^t) = \mathrm{KL}(\tau_B(x_B)||\tau_B^t(x_B)) = \sum_{x_B} \tau_B(x_B) \log \frac{\tau_B(x_B)}{\tau_B^t(x_B)}.$$

In this case, the proximal point algorithm reduces to Algorithm 4, which iteratively solves a smoothed free energy objective, with natural parameter θ^t updated at each iteration. Intuitively, the proximal inner loop (28)-(29) essentially "adds back" the truncated entropy term $H_B(\tau)$, while

canceling its effect by adjusting θ in the opposite direction. Typical choices of λ^t include $\lambda^t = 1$ (constant) and $\lambda^t = 1/t$ (harmonic). Note that the proximal approach is distinct from an annealing method, which would require that the annealing coefficient vanish to zero. Interestingly, if we take $\lambda^t = 1$, then the inner maximization problem (29) reduces to the standard log-partition function duality (1), corresponding to a pure marginalization task. This has the interpretation of transforming the marginal MAP problem into a sequence of standard sum-inference problems.

In practice we approximate $H_{A|B}(\tau)$ and $H_B(\tau)$ by pairwise entropy decomposition $\hat{H}_{A|B}(\tau)$ and $\hat{H}_B(\tau)$ in (15), respectively. If $\hat{H}_B(\tau)$ is provably convex in the sense of Weiss et al. (2007), that is, there exist positive constants $\{\kappa_i, \kappa_{i\to j}\}$ satisfying $\rho_i = \kappa_i + \sum_{k\in\partial_i} \kappa_{k\to i}$ and $\rho_{ij} = \kappa_{i\to j} + \kappa_{j\to i}$ for $i, j \in B$. Then the resulting approximate algorithm can be interpreted as a proximal algorithm that maximizes $\hat{F}_{mix}(\tau, \theta)$ with proximal function as

$$D_{pair}(\tau||\tau^t) = \sum_{i\in B} \kappa_i \mathrm{KL}[\tau_i(x_i)||\tau_i^0(x_i)] \; + \sum_{(ij)\in E_B} \kappa_{i\to j} \mathrm{KL}[(\tau_{ij}(x_i|x_j)||\tau_{ij}^0(x_i|x_j)].$$

In this case, Algorithm 4 is still a valid proximal algorithm and inherits its convergence guarantees. In practice one uses approximations that are not provably convex. An interesting special case is when both $H_{A|B}(\tau)$ and $H_B(\tau)$ are approximated by a Bethe approximation. This has the effect that the optimization (29) can be solved using standard belief propagation. Although the Bethe form for $H_{A|B}(\tau)$ and $H_B(\tau)$ is provably convex only in some special cases, such as when G is tree structured, we find in practice that this approximation gives very accurate solutions, even on general loopy graphs where its convergence is no longer theoretically guaranteed.

The global convergence guarantees of the proximal point algorithm may also fail if the inner update (29) is not solved exactly. It should also be possible to develop globally convergent algorithms without inner loops using the techniques that have been developed for full marginalization or MAP problems (e.g., Meltzer et al., 2009; Hazan and Shashua, 2010; Jojic et al., 2010; Savchynskyy et al., 2010), but we leave this to future work.

7. Connections to EM

A natural algorithm for solving the marginal MAP problem is to use the expectation-maximization (EM) algorithm, by treating x_A as the hidden variables and x_B as the "parameters" to be maximized. In this section, we show that the EM algorithm can be seen as a coordinate ascent algorithm on a mean field variant of our framework.

We start by introducing a "non-convex" generalization of Theorem 2.

Corollary 11. *Let $\mathbb{M}^o$ be the subset of the marginal polytope $\mathbb{M}$ corresponding to the distributions in which x_B are clamped to some deterministic values, that is,*

$$\mathbb{M}^o = \{\tau \in \mathbb{M} : \; \exists x_B^* \in \mathcal{X}_B, \text{ such that } \tau(x_B) = 1(x_B = x_B^*)\}.$$

Then the dual optimization (8) *remains exact if the marginal polytope $\mathbb{M}$ is replaced by any $\mathbb{N}$ satisfying $\mathbb{M}^o \subseteq \mathbb{N} \subseteq \mathbb{M}$, that is,*

$$\Phi_{AB} = \max_{\tau\in\mathbb{N}} \{\langle \theta, \tau\rangle + H_{A|B}(\tau)\}.$$

Proof. For an arbitrary marginal MAP solution x_B^*, the τ^* with $\tau^*(x) = p(x|x_B = x_B^*; \theta)$ is an optimum of (8) and satisfies $\tau^* \in \mathbb{M}^o$. Therefore, restricting the optimization on $\mathbb{M}^o$ (or any $\mathbb{N}$) does not change the maximum value of the objective function. □

Remark. Among all $\mathbb{N}$ satisfying $\mathbb{M}^o \subseteq \mathbb{N} \subseteq \mathbb{M}$, the marginal polytope $\mathbb{M}$ is the smallest (and the unique) convex set that includes $\mathbb{M}^o$, that is, it is the convex hull of $\mathbb{M}^o$.

To connect to EM, we define $\mathbb{M}^\times$, the set of distributions in which x_A and x_B are independent, that is, $\mathbb{M}^\times = \{\tau \in \mathbb{M} \colon \tau(x) = \tau(x_A)\tau(x_B)\}$. Since $\mathbb{M}^o \subset \mathbb{M}^\times \subset \mathbb{M}$, the dual optimization (8) remains exact when restricted to $\mathbb{M}^\times$, that is,

$$\Phi_{AB}(\theta) = \max_{\tau \in \mathbb{M}^\times} \{\langle \theta, \tau \rangle + H_{A|B}(\tau)\} = \max_{\tau \in \mathbb{M}^\times} \{\langle \theta, \tau \rangle + H_A(\tau)\},$$

where the second equality holds because $H_{A|B}(\tau) = H_A(\tau)$ for $\tau \in \mathbb{M}^\times$.

Although $\mathbb{M}^\times$ is no longer a convex set, it is natural to consider a coordinate update that alternately optimizes $\tau(x_A)$ and $\tau(x_B)$,

$$\begin{aligned} &\text{Updating sum part:} && \tau_A^{t+1} \leftarrow \operatorname*{argmax}_{\tau_A \in \mathbb{M}_A} \{\langle \mathbb{E}_{\tau_B^t}(\theta), \tau_A \rangle + H_A(\tau_A)\}, \\ &\text{Updating max part:} && \tau_B^{t+1} \leftarrow \operatorname*{argmax}_{\tau_B \in \mathbb{M}_B} \langle \mathbb{E}_{\tau_A^{t+1}}(\theta), \tau_B \rangle, \end{aligned} \tag{30}$$

where $\mathbb{M}_A$ and $\mathbb{M}_B$ are the marginal polytopes over x_A and x_B, respectively. Note that the sum and max step each happen to be the dual of a sum-inference and max-inference problem, respectively. If we go back to the primal, and update the primal configuration x_B instead of τ_B, (30) can be rewritten into

$$\begin{aligned} &\text{E step:} && \tau_A^{t+1}(x_A) \leftarrow p(x_A | x_B^t; \theta), \\ &\text{M step:} && x_B^{t+1} \leftarrow \arg\max_{x_B} \mathbb{E}_{\tau_A^{t+1}}(\theta), \end{aligned}$$

which is exactly the EM update, viewing x_B as parameters and x_A as hidden variables. Similar connections between EM and the coordinate ascent method on variational objectives has been discussed in Neal and Hinton (1998) and Wainwright and Jordan (2008).

When the E-step or M-step are intractable, one can insert various approximations. In particular, approximating $\mathbb{M}_A$ by a mean-field inner bound $\mathbb{M}_A^{mf}$ leads to variational EM. An interesting observation is obtained by using a Bethe approximation (3) to solve the E-step and a linear relaxation to solve the M-step; in this case, the EM-like update is equivalent to solving

$$\max_{\tau \in \mathbb{L}^\times} \Big\{ \langle \theta, \tau \rangle + \sum_{i \in A} H_i(\tau) \; - \sum_{(ij) \in E_A} I_{ij}(\tau) \Big\}, \tag{31}$$

where $\mathbb{L}^\times$ is the subset of $\mathbb{L}$ in which $\tau_{ij}(x_i, x_j) = \tau_i(x_i)\tau_j(x_j)$ for $(ij) \in \partial_{AB}$. Equivalently, $\mathbb{L}^\times$ is the subset of $\mathbb{L}$ in which $I_{ij}(\tau) = 0$ for $(ij) \in \partial_{AB}$. Therefore, (31) can be treated as a special case of (14) by taking $\rho_{ij} \to +\infty$, forcing the solution τ^* to fall into $\mathbb{L}^\times$. As we discussed in Section 4.3, EM represents an extreme of the tradeoff between convexity and integrality implied by Theorem 5, which strongly encourages vertex solutions by sacrificing convexity, and hence is likely to become stuck in local optima.

8. Junction Graph Belief Propagation for Marginal MAP

In the above, we have restricted the discussion to pairwise models and pairwise entropy approximations, mainly for the purpose of clarity. In this section, we extend our algorithms to leverage

higher order cliques, based on the junction graph representation (Mateescu et al., 2010; Koller and Friedman, 2009). Other higher order methods, like generalized BP (Yedidia et al., 2005) or their convex variants (Wainwright et al., 2005a; Wiegerinck, 2005), can be derived similarly.

For notation, a cluster graph is a graph of subsets of variables (called clusters). Formally, it is a triple $(\mathcal{G}, \mathcal{C}, \mathcal{S})$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is an undirected graph, with each node $k \in \mathcal{V}$ associated with a cluster $c_k \in \mathcal{C}$, and each edge $(kl) \in \mathcal{E}$ with a subset $s_{kl} \in \mathcal{S}$ (called separators) satisfying $s_{kl} \subseteq c_k \cap c_l$. We assume that $\mathcal{C}$ subsumes the index set I, that is, for any $\alpha \in I$, we can assign it with a $c_k \in \mathcal{C}$, denoted $c[\alpha]$, such that $\alpha \subseteq c_k$. In this case, we can reparameterize $\theta = \{\theta_\alpha : \alpha \in I\}$ into $\theta = \{\theta_{c_k} : k \in \mathcal{V}\}$ by taking $\theta_{c_k} = \sum_{\alpha:\, c[\alpha]=c_k} \theta_\alpha$, without changing the distribution. Therefore, we simply assume $\mathcal{C} = I$ in this paper without loss of generality. A cluster graph is called a *junction graph* if it satisfies the *running intersection property*—for each $i \in V$, the induced sub-graph consisting of the clusters and separators that include i is a connected tree. A junction graph is a junction tree if $\mathcal{G}$ is a tree.

To approximate the variational dual form, we first replace $\mathbb{M}$ with a higher order locally consistent polytope $\mathbb{L}(\mathcal{G})$, which is the set of local marginals $\tau = \{\tau_{c_k}, \tau_{s_{kl}} : k \in \mathcal{V}, (kl) \in \mathcal{E}\}$ that are consistent on the intersections of the clusters and separators, that is,

$$\mathbb{L}(\mathcal{G}) = \{\tau : \sum_{x_{c_k \setminus s_{kl}}} \tau_{c_k}(x_{c_k}) = \tau(x_{s_{kl}}), \tau_{c_k}(x_{c_k}) \geq 0, \text{for } \forall\, k \in \mathcal{V}, (kl) \in \mathcal{E}\}.$$

Clearly, we have $\mathbb{M} \subseteq \mathbb{L}(\mathcal{G})$ and that $\mathbb{L}(\mathcal{G})$ is tighter than the pairwise polytope $\mathbb{L}$ we used previously.

We then approximate the joint entropy term by a linear combination of the entropies over the clusters and separators,

$$H(\tau) \approx \sum_{k \in \mathcal{V}} H_{c_k}(\tau) - \sum_{(kl) \in \mathcal{E}} H_{s_{kl}}(\tau),$$

where $H_{c_k}(\tau)$ and $H_{s_{kl}}(\tau)$ are the entropy of the local marginals τ_{c_k} and $\tau_{s_{kl}}$, respectively. Further, we approximate $H_B(\tau)$ by a slightly more restrictive entropy decomposition,

$$H_B(\tau) \approx \sum_{k \in \mathcal{V}} H_{\pi_k}(\tau),$$

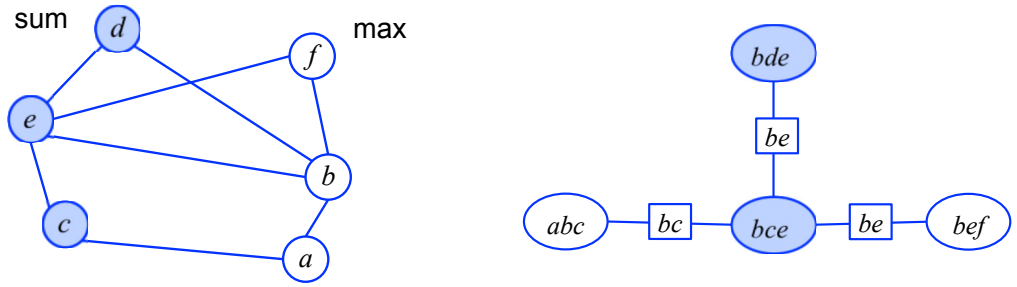
where $\{\pi_k : k \in \mathcal{V}\}$ is a non-overlapping partition of the max nodes B satisfying $\pi_k \subseteq c_k$ for $\forall k \in \mathcal{V}$. In other words, π represents an assignment of each max node $x_b \in B$ into a cluster k with $x_b \in \pi_k$. Let $\mathcal{B}$ be the set of clusters $k \in \mathcal{V}$ for which $\pi_k \neq \emptyset$, and call $\mathcal{B}$ the *max-clusters*; correspondingly, call $\mathcal{A} = \mathcal{V} \setminus \mathcal{B}$ the *sum-clusters*. See Figure 3 for an example.

Overall, the marginal MAP dual form in (8) is approximated by

$$\max_{\tau \in \mathbb{L}(\mathcal{G})} \big\{ \langle \theta, \tau \rangle + \sum_{k \in \mathcal{A}} H_{c_k}(\tau) + \sum_{k \in \mathcal{B}} H_{c_k|\pi_k}(\tau) - \sum_{(kl) \in \mathcal{E}} H_{s_{kl}}(\tau) \big\} \tag{32}$$

where $H_{c_k|\pi_k}(\tau) = H_{c_k}(\tau) - H_{\pi_k}(\tau)$. Optimizing (32) using a method similar to the derivation of mixed-product BP in Algorithm 2, we obtain a "mixed-product" junction graph belief propagation, given in Algorithm 5.

Similarly to our mixed-product BP in Algorithm 2, Algorithm 5 also admits an intuitive reparameterization interpretation and a strong local optimality guarantee. Algorithm 5 can be seen as



(a) (b)

Figure 3: (a) An example of marginal MAP problem, where d, c, e are sum nodes (shaded) and a, b, f are max nodes. (b) A junction graph of (a). Selecting a partitioning of max nodes, $\pi_{bde} = \pi_{bef} = \emptyset$, $\pi_{abc} = \{a, b\}$, and $\pi_{bef} = \{f\}$, results in $\{bde\}, \{bce\}$ being sum clusters (shaded) and $\{abc\}, \{bef\}$ being max clusters.

Algorithm 5 Mixed-product Junction Graph BP

1. Passing messages between clusters on the junction graph until convergence:

$\mathcal{A} \to \mathcal{A} \cup \mathcal{B}$: (sum-product)
$$m_{k \to l}(x_{s_{kl}}) \propto \sum_{x_{c_k \setminus s_{kl}}} \psi_{c_k}(x_{c_k}) m_{\sim k \setminus l}(x_{c_k}),$$

$\mathcal{B} \to \mathcal{A} \cup \mathcal{B}$: (argmax-product)
$$m_{k \to l}(x_{s_{kl}}) \propto \sum_{x_{c_k \setminus s_{kl}}} (\psi_{c_k}(x_{c_k}) m_{\sim k \setminus l}(x_{c_k})) \cdot 1[x_{\pi_k} \in \mathcal{X}^*_{\pi_k}],$$

$$\text{where } \mathcal{X}^*_{\pi_k} = \arg\max_{x_{\pi_k}} \sum_{x_{c_k \setminus \pi_k}} b_k(x_{c_k}),$$

$$b_k(x_{c_k}) = \psi_{c_k}(x_{c_k}) \prod_{k' \in \mathcal{N}(k)} m_{k' \to k}(x_{s_{k'k}}) \quad \text{and} \quad m_{\sim k \setminus l}(x_{c_k}) = \prod_{k' \in \mathcal{N}(k) \setminus \{l\}} m_{k' \to k}(x_{s_{k'k}}).$$

2. Decoding: $x^*_{\pi_k} = \arg\max_{x_{\pi_k}} \sum_{x_{c_k \setminus \pi_k}} b_k(x_{c_k})$ for $\forall k \in \mathcal{B}$.

a special case of a more general junction graph BP algorithm derived in Liu and Ihler (2012) for solving maximum expected utility tasks in decision networks. For more details, we refer the reader to that work.

9. Experiments

We illustrate our algorithms on both simulated models and more realistic diagnostic Bayesian networks taken from the UAI08 inference challenge. We show that our Bethe approximation algorithms perform best among all the tested algorithms, including Jiang et al. (2011)'s hybrid message passing and a state-of-the-art local search algorithm (Park and Darwiche, 2004).

We implement our mixed-product BP in Algorithm 2 with Bethe weights (`mix-product (Bethe)`), the regular sum-product BP (`sum-product`), max-product BP (`max-product`) and Jiang et al. (2011)'s hybrid message passing (with Bethe weights) in Algorithm 3 (`Jiang's method`),

where the solutions are all extracted by maximizing the singleton marginals of the max nodes. For all these algorithms, we run a maximum of 50 iterations; in case they fail to converge, we run 100 additional iterations with a damping coefficient of 0.1. We initialize all these algorithms with 5 random initializations and pick the best solution; for `mix-product (Bethe)` and `Jiang's method`, we run an additional trial initialized using the sum-product messages, which was reported to perform well in Park and Darwiche (2004) and Jiang et al. (2011). We also run the proximal point version of mixed-product BP with Bethe weights (`Proximal (Bethe)`), which is Algorithm 4 with both $H_{A|B}(\tau)$ and $H_B(\tau)$ approximated by Bethe approximations.

We also implement the TRW approximation, but only using the convergent proximal point algorithm, because the TRW upper bounds are valid only when the algorithms converge. The TRW weights of $\hat{H}_{A|B}$ are constructed by first (randomly) selecting spanning trees of G_A, and then augmenting each spanning tree with one uniformly selected edge in ∂_{AB}; the TRW weights of $\hat{H}_B(\tau)$ are constructed to be provably convex, using the method of TRW-S in Kolmogorov (2006). We run all the proximal point algorithms for a maximum of 100 iterations, with a maximum of 5 iterations of weighted message passing updates (18)-(19) for the inner loops (with 5 additional damping with 0.1 damping coefficient).

In addition, we compare our algorithms with SamIam, which is a state-of-the-art implementation of the local search algorithm for marginal MAP (Park and Darwiche, 2004); we use its default Taboo search method with a maximum of 500 searching steps, and report the best results among 5 trials with random initializations, and one additional trial initialized by its default method (which sequentially initializes x_i by maximizing $p(x_i|x_{\mathrm{pa}_i})$ along some predefined order).

We also implement an EM algorithm, whose expectation and maximization steps are approximated by sum-product and max-product BP, respectively. We run EM with 5 random initializations and one initialization by sum-product marginals, and pick the best solution.

9.1 Simulated Models

We consider pairwise models over discrete random variables taking values in $\{-1,0,+1\}^n$,

$$p(x) \propto \exp\big[\sum_i \theta_i(x_i) + \sum_{(ij)\in E} \theta_{ij}(x_i,x_j)\big].$$

The value tables of θ_i and θ_{ij} are randomly generated from normal distribution, $\theta_i(k) \sim \mathrm{Normal}(0,0.01)$, $\theta_{ij}(k,l) \sim \mathrm{Normal}(0,\sigma^2)$, where σ controls the strength of coupling. Our results are averaged on 1000 randomly generated sets of parameters.

We consider different choices of graph structures and max / sum node patterns:

1. *Hidden Markov chain* with 20 nodes, as shown in Figure 1.

2. *Latent tree models*. We generate random trees of size 50, by finding the minimum spanning trees of random symmetric matrices with elements drawn from $\mathrm{Uniform}([0,1])$. We take the leaf nodes to be max nodes, and the non-leaf nodes to be sum nodes. See Figure 5(a) for a typical example.

3. 10×10 *Grid* with max and sum nodes distributed in two opposite chess board patterns shown in Figure 6(a) and Figure 7(a), respectively. In Figure 6(a), the sum part is a loopy graph, and the max part is a (fully disconnected) tree; in Figure 7(a), the max and sum parts are flipped.

The results on the hidden Markov chain are shown in Figure 4, where we plot in panel (a) different algorithms' percentages of obtaining the globally optimal solutions among 1000 random trials, and in panel (b) their relative energy errors defined by $Q(\hat{x}_B;\theta) - Q(x_B^*;\theta)$, where $\hat{x}_B$ is the solution returned by the algorithms, and x_B^* is the true optimum.

The results of the latent tree models and the two types of 2D grids are shown in Figure 5, Figure 6 and Figure 7, respectively. Since the globally optimal solution x_B^* is not tractable to calculate in these cases, we report the approximate relative error defined by $Q(\hat{x}_B;\theta) - Q(\tilde{x}_B;\theta)$, where $\tilde{x}_B$ is the best solution we found across all algorithms.

9.2 Diagnostic Bayesian Networks

We also test our algorithms on two diagnostic Bayesian networks taken from the UAI08 Inference Challenge, where we construct marginal MAP problems by randomly selecting varying percentages of nodes to be max nodes. Since these models are not pairwise, we implement the junction graph versions of `mix-product (Bethe)` and `proximal (Bethe)` shown in Section 8. Figure 8 shows the approximate relative errors of our algorithms and `local search (SamIam)` as the percentage of the max nodes varies.

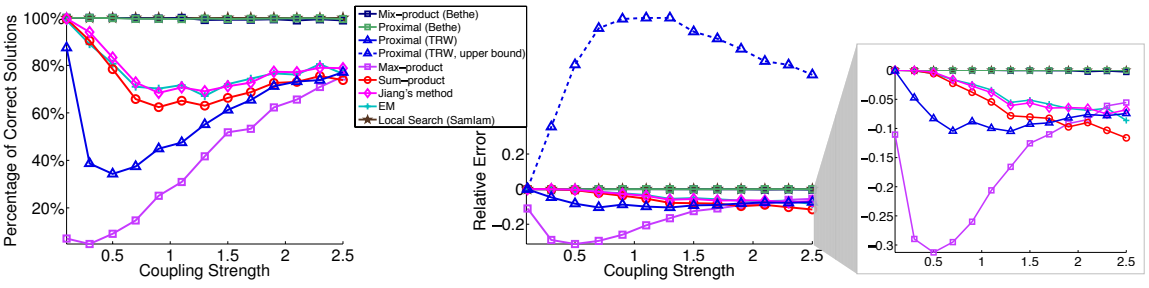
9.3 Insights

Across all the experiments, we find that `mix-product (Bethe)`, `proximal (Bethe)` and `local search (SamIam)` significantly outperform all the other algorithms, while `proximal (Bethe)` outperforms the two others in some circumstances. In the hidden Markov chain example in Figure 4, these three algorithms almost always (with probability $\geq 99\%$) find the globally optimal solutions. However, the performance of SamIam tends to degenerate when the max part has loopy dependency structures (see Figure 7), or when the number of max nodes is large (see Figure 8), both of which make it difficult to explore the solution space by local search. On the other hand, `mix-product (Bethe)` tends to degenerate as the coupling strength σ increases (see Figure 7), probably because its convergence gets worse as σ increases.

We note that our TRW approximation gives much less accurate solutions than the other algorithms, but is able to provide an upper bound on the optimal energy. Similar phenomena have been observed for TRW-BP in standard max- and sum- inference.

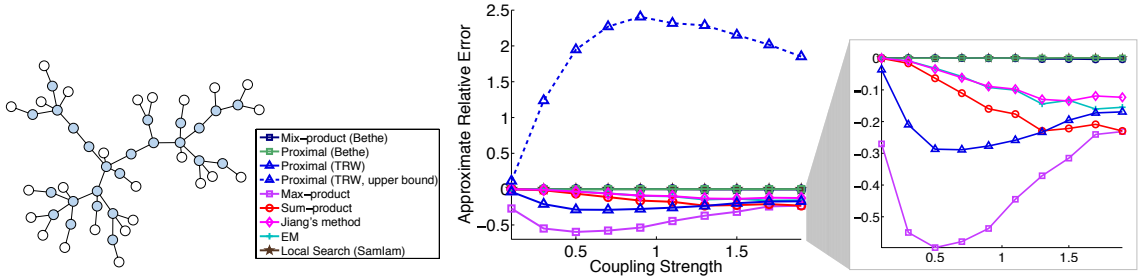
The hybrid message passing of Jiang et al. (2011) is significantly worse than `mix-product (Bethe)`, `proximal (Bethe)` and `local search (SamIam)`, but is otherwise the best among the remaining algorithms. EM performs similarly to (or sometimes worse than) Jiang's method.

The regular max-product BP and sum-product BP are among the worst of the tested algorithms, indicating the danger of approximating mixed-inference by pure max- or sum- inference. Interestingly, the performances of max-product BP and sum-product BP have opposite trends: In Figure 4, Figure 5 and Figure 6, where the max parts are fully disconnected and the sum parts are connected and loopy, max-product BP usually performs worse than sum-product BP, but gets better as the coupling strength σ increases; sum-product BP, on the other hand, tends to degenerate as σ increases. In Figure 7, where the max / sum pattern is reversed (resulting in a larger, loopier max subgraph), max-product BP performs better than sum-product BP.



(a) (b)

Figure 4: Results on the hidden Markov chain in Figure 1 (best viewed in color). (a) different algorithms' probabilities of obtaining the globally optimal solution among 1000 random trials. `Mix-product (Bethe)`, `Proximal (Bethe)` and `Local Search (SamIam)` almost always (with probability $\geq 99\%$) find the optimal solution. (b) The relative energy errors of the different algorithms, and the upper bounds obtained by `Proximal (TRW)` as a function of coupling strength σ.



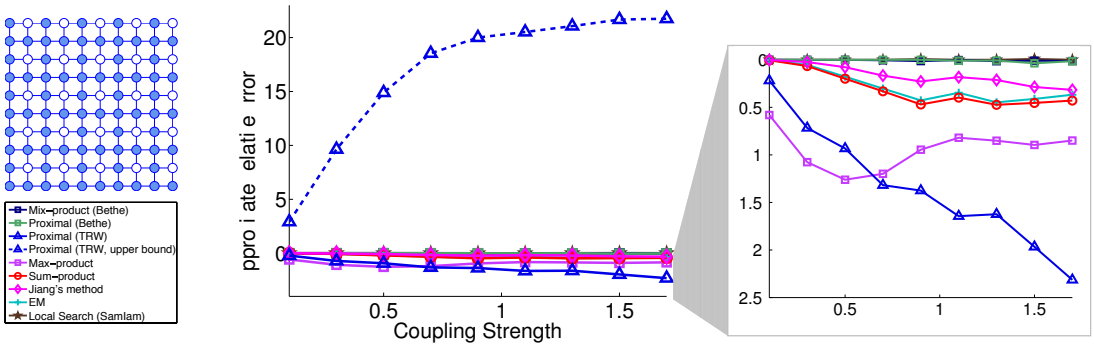
(a) (b)

Figure 5: (a) A typical latent tree model, whose leaf nodes are taken to be max nodes (white) and non-leaf nodes to be sum nodes (shaded). (b) The approximate relative energy errors of different algorithms, and the upper bound obtained by `Proximal (TRW)` as a function of coupling strength σ.

10. Conclusion and Further Directions

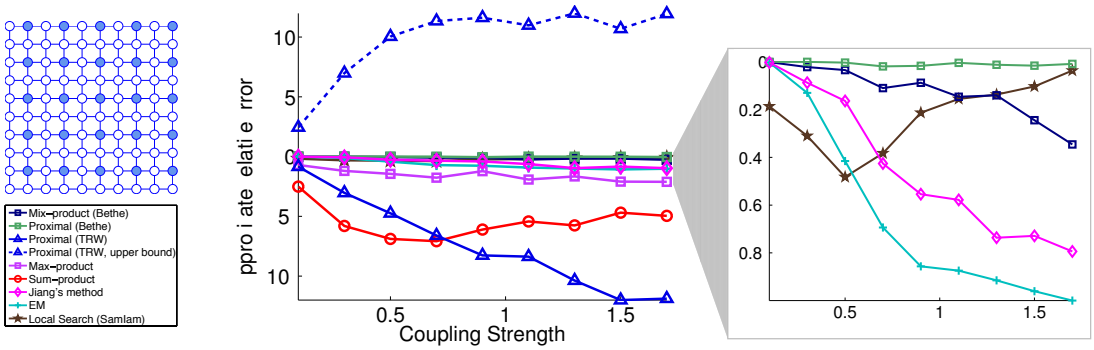
We have presented a general variational framework for solving marginal MAP problems approximately, opening new doors for developing efficient algorithms. In particular, we show that our proposed "mixed-product" BP admits appealing theoretical properties and performs well in practice.

Potential future directions include improving the performance of the truncated TRW approximation by optimizing weights, deriving optimality conditions that may be applicable even when the sum component does not form a tree, studying the convergent properties of mixed-product BP, and leveraging our results to learn hidden variable models for data.



(a) (b)

Figure 6: (a) A marginal MAP problem defined on a 10×10 Ising grid, with shaded sum nodes and unshaded max nodes; note that the sum part is a loopy graph, while max part is fully disconnected. (b) The approximate relative errors of different algorithms and the upper bound obtained by Proximal (TRW) as a function of coupling strength σ.

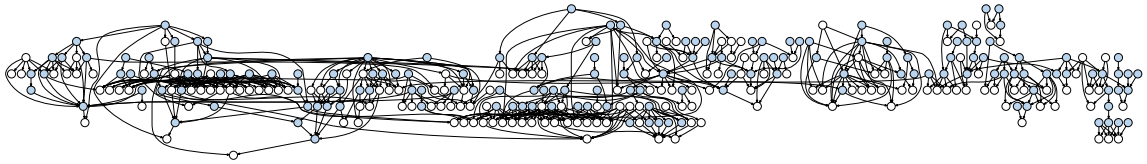


(a) (b)

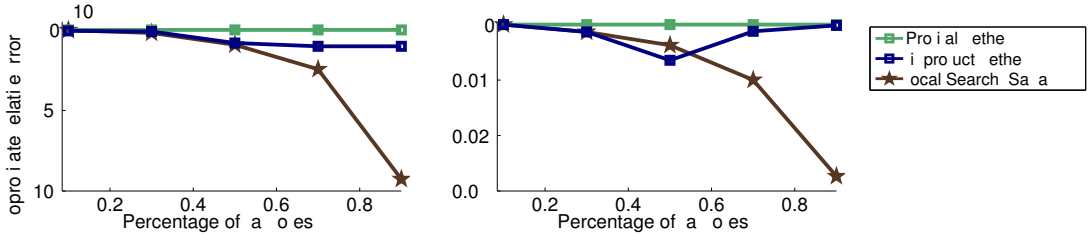
Figure 7: (a) A marginal MAP problem defined on a 10×10 Ising grid, but with max / sum part exactly opposite to that in Figure 6; note that the max part is loopy, while the sum part is fully disconnected in this case. (b) The approximate relative errors of different algorithms and the upper bound obtained by Proximal (TRW) as a function of coupling strength σ.

Acknowledgments

We thank Arthur Choi for providing help on SamIam. This work was supported in part by the National Science Foundation (awards IIS-1065618 and IIS-1254071), and a Microsoft Research Ph.D Fellowship.



(a) The structure of Diagnostic BN-2, with 50% randomly selected sum nodes shaded.



(b) Diagnostic BN-1 (c) Diagnostic BN-2

Figure 8: The results on two diagnostic Bayesian networks (BNs) in the UAI08 inference challenge. (a) The Diagnostic BN-2 network. (b)-(c) The performances of algorithms on the two BNs as a function of the percentage of max nodes. The local search method tends to degenerate when the number of max nodes is large, making it difficult to search over the solution space. Results are averaged over 100 random trials.

Appendix A. Proof of Proposition 6

Proof. The Lagrangian of (16) with the local consistency constraint of $\mathbb{L}$ in (2) is

$$\langle \theta, \tau \rangle + \sum_{i \in V} [w_i H_i(\tau) + \lambda_i^0 \sum_{x_i} \tau_i(x_i)] - \sum_{(ij) \in E} [w_{ij} I_{ij}(\tau) + \sum_{x_j} \lambda_{i \to j}(x_j) \sum_{x_i} (\tau_{ij}(x_i, x_j) - \tau_j(x_j))],$$

where $\{\lambda_i^0 \colon i \in V\}$ and $\{\lambda_{j \to i}(x_i) \colon (ij) \in E, x_i \in \mathcal{X}_i\}$ are the Lagrange multipliers. Recall that

$$\begin{aligned}
\langle \theta, \tau \rangle &= \sum_{i \in V} \theta_i(x_i) \tau_i(x_i) + \sum_{(ij) \in E} \theta_{ij}(x_i, x_j) \tau_{ij}(x_i, x_j), \\
H_i(\tau) &= -\sum_{x_i} \tau_i(x_i) \log \tau_i(x_i), \\
I_{ij}(\tau) &= \sum_{x_i, x_j} \tau_{ij}(x_i, x_j) \log \frac{\tau_{ij}(x_i, x_j)}{\sum_{x_i} \tau_{ij}(x_i, x_j) \sum_{x_j} \tau_{ij}(x_i, x_j)}.
\end{aligned}$$

Taking the derivative of the Lagrangian w.r.t. $\tau_i(x_i)$ and $\tau_{ij}(x_i, x_j)$, we have

$$\theta_i(x_i) - w_i \log \tau_i(x_i) + \sum_{j \in \partial_i} \lambda_{j \to i}(x_i) = const, \tag{33}$$

$$\theta_{ij}(x_i, x_j) - w_{ij} \log \frac{\tau_{ij}(x_i, x_j)}{\tau_i(x_i) \tau_j(x_j)} + \lambda_{i \to j}(x_j) + \lambda_{j \to i}(x_i) = const, \tag{34}$$

where we used the local consistency condition that $\sum_{x_j} \tau_{ij}(x_i, x_j) = \tau_i(x_i)$. By defining $m_{i \to j}(x_j) = \exp(\lambda_{i \to j}(x_j))$, we obtain (19) directly from (33)-(34).
Plugging (19) into the constraint that $\sum_{x_j} \tau_{ij}(x_i, x_j) = \tau_i(x_i)$ gives (18). □

Appendix B. Proof of Theorem 5

Proof. (i). For $\tau \in \mathbb{M}^o$, the objective function in (14) equals

$$\begin{aligned} F_{tree}(\tau,\theta) &= \langle \theta,\tau \rangle + \sum_{i\in V} H_i(\tau) - \sum_{(ij)\in E_A} I_{ij}(\tau) - \sum_{(ij)\in \partial_{AB}} \rho_{ij} I_{ij}(\tau) \\ &= \langle \theta,\tau \rangle + \sum_{i\in V} H_i(\tau) - \sum_{(ij)\in E_A} I_{ij}(\tau) \qquad (35) \\ &= \langle \theta,\tau \rangle + H_{A|B}(\tau) \qquad (36) \\ &= F_{mix}(\tau,\theta), \end{aligned}$$

where the equality in (35) is because $I_{ij}(\tau) = 0$ if $\forall (ij) \in \partial_{AB}$, and the equality in (36) is because the sum part G_A is a tree and we have the tree decomposition $H_{A|B} = \sum_{i\in V} H_i(\tau) - \sum_{(ij)\in E_A} I_{ij}(\tau)$. Therefore we have

$$\Phi_{tree}(\theta) = \max_{\tau\in\mathbb{L}} F_{tree}(\tau,\theta) \geq \max_{\tau\in\mathbb{M}^o} F_{tree}(\tau,\theta) = \max_{\tau\in\mathbb{M}^o} F_{mix}(\tau,\theta) = \Phi_{AB}(\theta), \qquad (37)$$

where the inequality is because $\mathbb{M}^o \subset \mathbb{M} \subset \mathbb{L}$.

If there exists x_B^* such that $Q(x_B^*;\theta) = \Phi_{tree}(\theta)$, then we have

$$Q(x_B^*;\theta) = \Phi_{tree}(\theta) \geq \Phi_{AB}(\theta) = \max_{x_B} Q(x_B;\theta).$$

This proves that x_B^* is a globally optimal marginal MAP solution.

(ii). Because $\tau_i^*(x_i)$ for $\forall i \in B$ are deterministic, and the sum part G_A is a tree, we have that $\tau^* \in \mathbb{M}^o$. Therefore the inequality in (37) is tight, and we can conclude the proof by using Corollary 11. □

Appendix C. Proof of Theorem 10

Proof. By Theorem 9, the beliefs $\{b_i, b_{ij}\}$ should satisfy the reparameterization property in (24) and the consistency conditions in (25)-(27). Without loss of generality, we assume $\{b_i, b_{ij}\}$ are normalized such that $\sum_{x_i} b_i(x_i) = 1$ for $i \in A$ and $\max_{x_i} b_i(x_i) = 1$ for $i \in B$.

I) For simplicity, we first prove the case of $C = B$, when $G = G_{C\cup A}$ itself is a semi A-B tree, and the theorem implies that x_B^* is a global optimum. By the reparameterization condition, we have

$$p(x) = \hat{p}_B(x_B)\hat{p}_{A|B}(x),$$

where

$$\hat{p}_B(x_B) = \prod_{i\in B} b_i(x_i) \prod_{(ij)\in E_B} \left[\frac{b_{ij}(x_i,x_j)}{b_i(x_i)b_j(x_j)}\right]^{\rho_{ij}}, \qquad (38)$$

$$\hat{p}_{A|B}(x) = \prod_{i\in A} b_i(x_i) \prod_{(ij)\in E_A} \left[\frac{b_{ij}(x_i,x_j)}{b_i(x_i)b_j(x_j)}\right]^{\rho_{ij}} \prod_{(ij)\in \partial_{AB}} \left[\frac{b_{ij}(x_i,x_j)}{b_i(x_i)b_j(x_j)}\right]^{\rho_{ij}}. \qquad (39)$$

Note we have

$$p(x_B) = \sum_{x_A} p(x) = \sum_{x_A} \hat{p}_B(x_B)\hat{p}_{A|B}(x) = \hat{p}_B(x_B) \sum_{x_A} \hat{p}_{A|B}(x).$$

We just need to show that x_B^* maximizes $\hat{p}_B(x_B)$ and $\sum_{x_A}\hat{p}_{A|B}(x)$, respectively.

First, since $\hat{p}_B(x_B)$ involves only the max nodes, a standard MAP analysis applies. Because the max part of the beliefs, $\{b_i, b_{ij}\colon (ij)\in E_B\}$, satisfy the standard max-consistency conditions, and the corresponding TRW weights $\{\rho_{ij}\colon (ij)\in E_B\}$ are provably convex by assumption, we establish that x_B^* is the MAP solution of $\hat{p}_B(x_B)$ by Theorem 1 of Weiss et al. (2007).

Secondly, to show that x_B^* also maximizes $\hat{p}_{A|B}(x)$ requires the combination of the mixed-consistency and sum-consistency conditions. Since G is a semi A-B tree, we denote by π_i the unique parent node of i ($\pi_i=\emptyset$ if i is a root). In addition, let ∂_A be the subset of A whose parent nodes are in B, that is, $\partial_A=\{i\in A\colon \pi_i\in B\}$. Equation (39) can be reformed into

$$\hat{p}_{A|B}(x)=\prod_{i\in A\setminus\partial_A}\frac{b_{i,\pi_i}(x_i,x_{\pi_i})}{b_{\pi_i}(x_{\pi_i})}\prod_{i\in\partial_A}\left[\frac{b_{i,\pi_i}(x_i,x_{\pi_i})}{b_{\pi_i}(x_{\pi_i})}\right]^{\rho_{i,\pi_i}}\left[b_i(x_i)\right]^{1-\rho_{i,\pi_i}},$$

where we used the fact that $\rho_{ij}=1$ for $(ij)\in E_A$. Therefore, we have for any $x_B\in\mathcal{X}_B$,

$$\begin{aligned}\sum_{x_A}\hat{p}_{A|B}(x)&=\sum_{x_A}\left\{\prod_{i\in A\setminus\partial_A}\frac{b_{i,\pi_i}(x_i,x_{\pi_i})}{b_{\pi_i}(x_{\pi_i})}\prod_{i\in\partial_A}\left[\frac{b_{i,\pi_i}(x_i,x_{\pi_i})}{b_{\pi_i}(x_{\pi_i})}\right]^{\rho_{i,\pi_i}}\left[b_i(x_i)\right]^{1-\rho_{i,\pi_i}}\right\}\\&=\prod_{i\in\partial_A}\sum_{x_i}\left[\frac{b_{i,\pi_i}(x_i,x_{\pi_i})}{b_{\pi_i}(x_{\pi_i})}\right]^{\rho_{i,\pi_i}}\left[b_i(x_i)\right]^{1-\rho_{i,\pi_i}}\qquad(40)\\&\le\prod_{i\in\partial_A}\left[\sum_{x_i}\frac{b_{i,\pi_i}(x_i,x_{\pi_i})}{b_{\pi_i}(x_{\pi_i})}\right]^{\rho_{i,\pi_i}}\left[\sum_{x_i}b_i(x_i)\right]^{1-\rho_{i,\pi_i}}\qquad(41)\\&=1,\qquad(42)\end{aligned}$$

where the equality in (40) eliminates (by summation) all the interior nodes in A. The inequality in (41) follows from Hölder's inequality. Finally, the equality in (42) holds because all the sum part of beliefs $\{b_i, b_{ij}\colon (ij)\in E_A\}$ satisfies the sum-consistency (25).

On the other hand, for any $(i,\pi_i)\in\partial_{AB}$, because $x_{\pi_i}^*=\arg\max_{x_{\pi_i}}b_{\pi_i}(x_{\pi_i})$, we have $b_{i,\pi_i}(x_i,x_{\pi_i}^*)=b_i(x_i)$ by the mixed-consistency condition (27). Therefore,

$$\begin{aligned}\sum_{x_A}\hat{p}_{A|B}([x_A,x_B^*])&=\prod_{i\in\partial_A}\sum_{x_i}\left[\frac{b_{i,\pi_i}(x_i,x_{\pi_i}^*)}{b_{\pi_i}(x_{\pi_i}^*)}\right]^{\rho_{i,\pi_i}}\left[b_i(x_i)\right]^{1-\rho_{i,\pi_i}}\\&=\prod_{i\in\partial_A}\left[\frac{1}{b_{\pi_i}(x_{\pi_i}^*)}\right]^{\rho_{i,\pi_i}}\sum_{x_i}b_i(x_i)\\&=1.\qquad(43)\end{aligned}$$

Combining (42) and (43), we have $\sum_{x_A}\hat{p}_{A|B}(x)\le\sum_{x_A}\hat{p}_{A|B}([x_A,x_B^*])=1$ for any $x_B\in\mathcal{X}_B$, that is, x_B^* maximizes $\sum_{x_A}\hat{p}_{A|B}(x)$. This finishes the proof for the case $C=B$.

II) In the case of $C\neq B$, let $D=B\setminus C$. We decompose $p(x)$ into

$$p(x)=\hat{p}_B([x_C,x_D])\hat{p}_{A|C}([x_A,x_C])\hat{r}_{AD}([x_A,x_D])$$

where $\hat{p}_B(x_B)$ and $\hat{p}_{A|B}(x)$ are defined similarly to (38) and (39),

$$\hat{p}_B(x_B) = \prod_{i\in B} b_i(x_i) \prod_{(ij)\in E_B} \left[\frac{b_{ij}(x_i,x_j)}{b_i(x_i)b_j(x_j)}\right]^{\rho_{ij}},$$
$$\hat{p}_{A|C}([x_A,x_C]) = \prod_{i\in A} b_i(x_i) \prod_{(ij)\in E_A} \left[\frac{b_{ij}(x_i,x_j)}{b_i(x_i)b_j(x_j)}\right]^{\rho_{ij}} \prod_{(ij)\in \partial_{AC}} \left[\frac{b_{ij}(x_i,x_j)}{b_i(x_i)b_j(x_j)}\right]^{\rho_{ij}},$$

where π_i is the parent node of i in the semi A-B tree $G_{A\cup C}$ and ∂_{AC} is set of edges across A and C, that is, $\partial_{AC} = \{(ij)\in E\colon i\in A, j\in C\}$. The term $\hat{r}_{AD}(x)$ is defined as

$$\hat{r}_{AD}([x_A,x_D]) = \prod_{(ij)\in\partial_{AD}} \left[\frac{b_{ij}(x_i,x_j)}{b_i(x_i)b_j(x_j)}\right]^{\rho_{ij}},$$

where similarly ∂_{AD} is the set of edges across A and D.

Because $x_j^* = \arg\max_{x_j} b_j(x_j)$ for $j\in D$, we have $b_{ij}(x_i,x_j^*) = b_i(x_i)$ for $(ij)\in\partial_{AD}$, $j\in D$ by the mixed-consistency condition in (27). Therefore, one can show that $\hat{r}_{AD}([x_A,x_D^*]) = 1$, and hence

$$p([x_A,x_C,x_D^*]) = \hat{p}_B([x_C,x_D^*])\hat{p}_{A|C}([x_A,x_C]).$$

The remainder of the proof is similar to that for the case $C = B$: by the analysis in Weiss et al. (2007), it follows that $x_C^* \in \arg\max_{x_C} p([x_C,x_D^*])$, and we have previously shown that $x_C^* \in \arg\max_{x_C} \sum_{x_A} \hat{p}_{A|C}([x_A,x_C])$. This establishes that x_C^* maximizes

$$\sum_{x_A} p([x_A,x_C,x_D^*]) = p([x_C,x_D^*]) \sum_{x_A} \hat{p}_{A|C}([x_A,x_C]),$$

which concludes the proof. □

References

F. Altarelli, A. Braunstein, A. Ramezanpour, and R. Zecchina. Stochastic optimization by message passing. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11):P11009, 2011.

J.R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Verlag, 1997.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 2010.

T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley & Sons, INC., 2nd edition, 2006.

C.P. De Campos. New complexity results for MAP in Bayesian networks. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 2100–2106, 2011.

R. Dechter and I. Rish. Mini-buckets: A general scheme for bounded inference. *Journal of the ACM (JACM)*, 50(2):107–153, 2003.

A. Doucet, S.J. Godsill, and C.P. Robert. Marginal maximum a posteriori estimation using Markov chain Monte Carlo. *Statistics and Computing*, 12(1), January 2002.

A. Globerson and T.S. Jaakkola. Approximate inference using conditional entropy decompositions. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07)*, 2007.

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 (Methodological)*, pages 271–279, 1989.

T. Hazan and A. Shashua. Norm-product belief propagation: Primal-dual message-passing for approximate inference. *Information Theory, IEEE Transactions on*, 56(12):6294–6316, 2010.

T. Hazan, J. Peng, and A. Shashua. Tightening fractional covering upper bounds on the partition function for high-order region graphs. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI-12)*, 2012.

R.A. Howard and J.E. Matheson. Influence diagrams. *Decision Analysis*, 2(3):127–143, 2005.

D.R. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 1(58), February 2004.

M. Ibrahimi, A. Javanmard, Y. Kanoria, and A. Montanari. Robust max-product belief propagation. In *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*, pages 43–49. IEEE, 2011.

A. Iusem and M. Teboulle. On the convergence rate of entropic proximal optimization methods. *Computational and Applied Mathematics*, 12:153–168, 1993.

E.T. Jaynes. Information theory and statistical mechanics. *Physical Review*, 106(4):620–630, May 1957.

M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the ising model. *SIAM Journal on Computing*, 22(5):1087–1116, 1993.

J. Jiang, P. Rai, and H. Daumé III. Message-passing for approximate map inference with latent variables. In *Advances in Neural Information Processing Systems (NIPS-11)*, 2011.

V. Jojic, S. Gould, and D. Koller. Accelerated dual decomposition for MAP inference. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009.

V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1568 –1583, oct. 2006.

Q. Liu and A. Ihler. Bounding the partition function using Hölder's inequality. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 849–856, June 2011a.

Q. Liu and A. Ihler. Variational algorithms for marginal MAP. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI-11)*. 2011b.

Q. Liu and A. Ihler. Belief propagation for structured decision making. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI-12)*. August 2012.

B. Martinet. Régularisation d'inéquations variationnelles par approximations successives. *Revue Française dInformatique et de Recherche Opérationelle*, 4:154–158, 1970.

R. Mateescu, K. Kask, V. Gogate, and R. Dechter. Join-graph propagation algorithms. *Journal of Artificial Intelligence Research*, 37(1):279–328, 2010.

D.D. Mauá and C.P. de Campos. Anytime marginal maximum a posteriori inference. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 2012.

T. Meltzer, A. Globerson, and Y. Weiss. Convergent message passing algorithms: a unifying view. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI-09)*, 2009.

O.P. Meshi, T. Jaakkola, and A. Globerson. Convergence rate analysis of MAP coordinate minimization algorithms. In *Advances in Neural Information Processing Systems (NIPS-12)*, 2012.

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

J. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research*, 21:101–133, 2004.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

P. Ravikumar, A. Agarwal, and M.J. Wainwright. Message-passing for graph-structured linear programs: Proximal projections, convergence, and rounding schemes. *Journal of Machine Learning Research*, 11:1043–1080, Mar 2010.

R.T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5):877, 1976.

B. Savchynskyy, S. Schmidt, J.H. Kappes, and C. Schnörr. Efficient MRF energy minimization via adaptive diminishing smoothing. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI-12)*, 2010.

D. Sontag, A. Globerson, and T.S. Jaakkola. Introduction to dual decomposition for inference. In *Optimization for Machine Learning*. MIT Press, 2011.

M. Teboulle. Entropic proximal mappings with applications to nonlinear programming. *Mathematics of Operations Research*, 17(3):pp. 670–690, 1992.

P. Tseng and D.P. Bertsekas. On the convergence of the exponential multiplier method for convex programming. *Mathematical Programming*, 60(1):1–19, 1993.

M.J. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundation and Trends in Machine Learning*, 1(1-2):1–305, 2008.

M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *Information Theory, IEEE Transactions on*, 45: 1120–1146, 2003.

M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. A new class of upper bounds on the log partition function. *Information Theory, IEEE Transactions on*, 51(7):2313–2335, July 2005a.

M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. MAP estimation via agreement on trees: message-passing and linear programming. *Information Theory, IEEE Transactions on*, 51(11): 3697–3717, 2005b.

Y. Weiss, C. Yanover, and T. Meltzer. MAP estimation, linear programming and belief propagation with convex free energies. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI-07)*, 2007.

T. Werner. A linear programming approach to max-sum problem: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(7):1165–1179, 2007.

T. Werner. Revisiting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(8):1474–1488, 2010.

W. Wiegerinck. Approximations with reweighted generalized belief propagation. In *In Proceedings of the 9th International Conference on Artificial Intelligence and Statistics (AISTATS-05)*, 2005.

J.S. Yedidia, W.T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring Artificial Intelligence in the New Millennium*, 8:236–239, 2003.

J.S. Yedidia, W.T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized BP algorithms. *Information Theory, IEEE Transactions on*, 51, July 2005.

C. Yuan, T.C. Lu, and M.J. Druzdzel. Annealed MAP. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 628–635, 2004.

A. L. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14(7):1691–1722, July 2002.

Journal of Machine Learning Research 14 (2013) 3201-3205 Submitted 1/12; Revised 2/13; Published 10/13

GURLS: A Least Squares Library for Supervised Learning

Andrea Tacchetti[*] ATACCHET@MIT.EDU
Pavan K. Mallapragada PAVAN_M@MIT.EDU
Center for Biological and Computational Learning, McGovern Institute for Brain Research
Massachusetts Institute of Technology, Bldg. 46-5155
Cambridge, MA, 02139, USA

Matteo Santoro MATTEO.SANTORO@IIT.IT
Laboratory for Computational and Statistical Learning, Istituto Italiano di Tecnologia
Via Morego, 30
16163 Genova, Italy

Lorenzo Rosasco[*] LROSASCO@MIT.EDU
DIBRIS, Università degli Studi di Genova
via Dodecaneso, 35
16146 Genova, Italy

Editor: Geoff Holmes

Abstract

We present GURLS, a least squares, modular, easy-to-extend software library for efficient supervised learning. GURLS is targeted to machine learning practitioners, as well as non-specialists. It offers a number state-of-the-art training strategies for medium and large-scale learning, and routines for efficient model selection. The library is particularly well suited for multi-output problems (multi-category/multi-label). GURLS is currently available in two independent implementations: Matlab and C++. It takes advantage of the favorable properties of regularized least squares algorithm to exploit advanced tools in linear algebra. Routines to handle computations with very large matrices by means of memory-mapped storage and distributed task execution are available. The package is distributed under the BSD license and is available for download at `https://github.com/LCSL/GURLS`.

Keywords: regularized least squares, big data, linear algebra

1. Introduction and Design

Supervised learning has become a fundamental tool for the design of intelligent systems and the analysis of high dimensional data. Key to this success has been the availability of efficient, easy-to-use software packages. New data collection technologies make it easy to gather high dimensional, multi-output data sets of increasing size. This trend calls for new software solutions for the automatic training, tuning and testing of supervised learning methods. These observations motivated the design of GURLS (Grand Unified Regularized Least Squares). The package was developed to pursue the following goals: *Speed*: Fast training/testing procedures for learning problems with potentially large/huge number of points, features and especially outputs (e.g., classes). *Memory*: Flexible data management to work with large data sets by means of memory-mapped storage. *Performance*:

∗. Also in the Laboratory for Computational and Statistical Learning, Istituto Italiano di Tecnologia and Massachusetts Institute of Technology

State of the art results in high-dimensional multi-output problems. *Usability and modularity*: Easy to use and to expand. GURLS is based on Regularized Least Squares (RLS) and takes advantage of all the favorable properties of these methods (Rifkin et al., 2003). Since the algorithm reduces to solving a linear system, GURLS is set up to exploit the powerful tools, and recent advances, of linear algebra (including randomized solver, first order methods, etc.). Second, it makes use of RLS properties which are particularly suited for high dimensional learning. For example: (1) RLS has natural primal and dual formulation (hence having complexity which is the smallest between number of examples and features); (2) efficient parameter selection (closed form expression of the leave one out error and efficient computations of regularization path); (3) natural and efficient extension to multiple outputs. Specific attention has been devoted to handle large high dimensional data sets. We rely on data structures that can be serialized using memory-mapped files, and on a distributed task manager to perform a number of key steps (such as matrix multiplication) without loading the whole data set in memory. Efforts were devoted to to provide a lean API and an exhaustive documentation. GURLS has been deployed and tested successfully on Linux, MacOS and Windows. The library is distributed under the simplified BSD license, and can be downloaded from `https://github.com/LCSL/GURLS`.

2. Description of the Library

The library comprises four main modules. GURLS and bGURLS—both implemented in Matlab—are aimed at solving learning problems with small/medium and large-scale data sets respectively. GURLS^{++}and bGURLS^{++}are their C++ counterparts. The Matlab and C++ versions share the same design, but the C^{++}modules have significant improvements, which make them faster and more flexible. The specification of the desired machine learning experiment in the library is straightforward. Basically, it is a formal description of a *pipeline*, that is, an ordered sequence of steps. Each step identifies an actual learning task, and belongs to a predefined category. The core of the library is a method (a class in the C^{++}implementation) called *GURLScore*, which is responsible for processing the sequence of tasks in the proper order and for linking the output of the former task to the input of the subsequent one. A key role is played by the additional "options" structure, referred to as OPT. OPT is used to store all configuration parameters required to customize the behavior of individual tasks in the pipeline. Tasks receive configuration parameters from OPT in read-only mode and—upon termination—the results are appended to the structure by *GURLScore* in order to make them available to subsequent tasks. This allows the user to skip the execution of some tasks in a pipeline, by simply inserting the desired results directly into the options structure. Currently, we identify six different task categories: data set splitting, kernel computation, model selection, training, evaluation and testing and performance assessment and analysis. Tasks belonging to the same category may be interchanged with each other.

2.1 Learning From Large Data Sets

Two modules in GURLS have been specifically designed to deal with *big data* scenarios. The approach we adopted is mainly based on a memory-mapped abstraction of matrix and vector data structures, and on a distributed computation of a number of standard problems in linear algebra. For learning on big data, we decided to focus specifically on those situations where one seeks a linear model on a large set of (possibly non linear) features. A more accurate specification of what "large" means in GURLS is related to the number of features d and the number of training

data set	# of samples	# of classes	# of variables
optdigit	3800	10	64
landast	4400	6	36
pendigit	7400	10	16
letter	10000	26	16
isolet	6200	26	600

Table 1: Data sets description.

examples n: we require it must be possible to store a $\min(d,n) \times \min(d,n)$ matrix in memory. In practice, this roughly means we can train models with up-to $25k$ features on machines with $8Gb$ of RAM, and up-to $50k$ features on machines with $36Gb$ of RAM. We *do not* require the data matrix itself to be stored in memory: within GURLS it is possible to manage an arbitrarily large set of training examples. We distinguish two different scenarios. Data sets that can fully reside in RAM without any memory mapping techniques—such as swapping—are considered to be small/medium. Larger data sets are considered to be "big" and learning must be performed using either bGURLS or bGURLS^{++}. These two modules include all the design patterns described above, and have been complemented with additional big data and distributed computation capabilities. Big data support is obtained using a data structure called *bigarray*, which allows to handle data matrices as large as the space available on the hard drive: we store the entire data set on disk and load only small chunks in memory when required. There are some differences between the Matlab and C^{++}implementations. bGURLS relies on a simple, ad hoc interface, called *GURLS Distributed Manager* (GDM), to distribute matrix-matrix multiplications, thus allowing users to perform the important task of kernel matrix computation on a distributed network of computing nodes. After this step, the subsequent tasks behave as in GURLS. bGURLS^{++}(currently in active development) offers more interesting features because it is based on the MPI libraries. Therefore, it allows for a full distribution within every single task of the pipeline. All the processes read the input data from a shared filesystem over the network and then start executing the same pipeline. During execution, each process' task communicates with the corresponding ones. Every process maintains its local copy of the options. Once the same task is completed by all processes, the local copies of the options are synchronized. This architecture allows for the creation of hybrid pipelines comprising serial one-process-based tasks from GURLS^{++}.

3. Experiments

We decided to focus the experimental analysis in the paper to the assessment of GURLS' performance both in terms of accuracy and time. In our experiments we considered 5 popular data sets, briefly described in Table 1. Experiments were run on a Intel Xeon 5140 @ 2.33GHz processor with 8GB of RAM, and running Ubuntu 8.10 Server (64 bit).

	optdigit		**landsat**		**pendigit**	
	accuracy (%)	time (s)	accuracy (%)	time (s)	accuracy (%)	time (s)
GURLS (linear primal)	92.3	0.49	63.68	0.22	82.24	0.23
GURLS (linear dual)	92.3	726	66.3	1148	82.46	5590
LS-SVM linear	92.3	7190	64.6	6526	82.3	46240
GURLS (500 random features)	96.8	25.6	63.5	28.0	96.7	31.6
GURLS (1000 random features)	97.5	207	63.5	187	95.8	199
GURLS (Gaussian kernel)	98.3	13500	90.4	20796	98.4	100600
LS-SVM (Gaussian kernel)	98.3	26100	90.51	18430	98.36	120170

Table 2: Comparison between GURLS and LS-SVM.

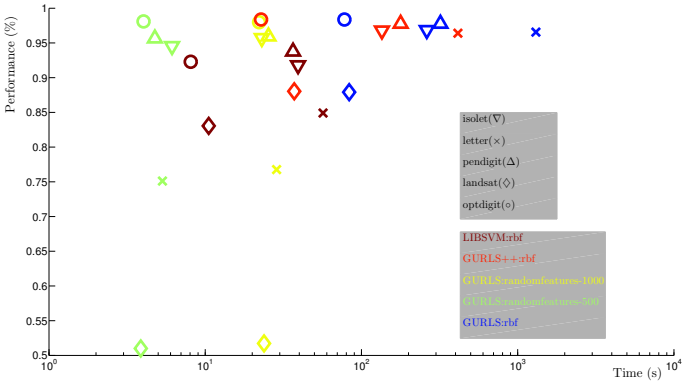


Figure 1: Prediction accuracy vs. computing time. The color represents the training method and the library used. In blue: the Matlab implementation of RLS with RBF kernel, in red: its C++ counterpart. In dark red: results of LIBSVM with RBF kernel. In yellow and green: results obtained using a linear kernel on 500 and 1000 random features respectively.

We set up different pipelines and compared the performance to SVM, for which we used the python modular interface to LIBSVM (Chang and Lin, 2011). Automatic selection of the optimal regularization parameter is implemented identically in all experiments: (i) split the data; (ii) define a set of regularization parameter on a regular grid; (iii) perform hold-out validation. The variance of the Gaussian kernel has been fixed by looking at the statistics of the pairwise distances among training examples. The prediction accuracy of GURLS and GURLS^{++}is identical—as expected—but the implementation in C++ is significantly faster. The prediction accuracy of standard RLS-based methods is in many cases higher than SVM. Exploiting the primal formulation of RLS, we further ran experiments with the random features approximation (Rahimi and Recht, 2008). As show in Figure 1, the performance of this method is comparable to that of SVM at a much lower computational cost in the majority of the tested data sets. We further compared GURLS with another available least squares based toolbox: the LS-SVM toolbox (Suykens et al., 2001), which includes routines for parameter selection such as *coupled simulated annealing* and line/grid search. The goal of this experiment is to benchmark the performance of the parameter selection with random data splitting included in GURLS. For a fair comparison, we considered only the Matlab implementation of GURLS. Results are reported in Table 2. As expected, using the linear kernel with the primal formulation—not available in LS-SVM—is the fastest approach since it leverages the lower dimensionality of the input space. When the Gaussian kernel is used, GURLS and LS-SVM have comparable computing time and classification performance. Note, however, that in GURLS the number of parameter in the grid search is fixed to 400, while in LS-SVM it may vary and is limited to 70. The interesting results obtained with the random features implementation in GURLS, make it an interesting choice in many applications. Finally, all GURLS pipelines, in their Matlab implementation, are faster than LS-SVM and further improvements can be achieved with GURLS^{++}.

Acknowledgments

We thank Tomaso Poggio, Zak Stone, Nicolas Pinto, Hristo S. Paskov and CBCL for comments and insights.

References

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems*, volume 21, pages 1313–1320, 2008.

R. Rifkin, G. Yeo, and T. Poggio. Regularized least-squares classification. *Nato Science Series Sub Series III Computer and Systems Sciences*, 190:131–154, 2003.

J. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, 2001. ISBN 981-238-151-1.

Journal of Machine Learning Research 14 (2013) 3207-3260 Submitted 9/12; Revised 3/13; Published 11/13

Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising

Léon Bottou LEON@BOTTOU.ORG
Microsoft
1 Microsoft Way
Redmond, WA 98052, USA

Jonas Peters* PETERS@STAT.MATH.ETHZ.CH
Max Planck Institute
Spemannstraße 38
72076 Tübingen, Germany

Joaquin Quiñonero-Candela† JQUINONERO@GMAIL.COM
Denis X. Charles CDX@MICROSOFT.COM
D. Max Chickering DMAX@MICROSOFT.COM
Elon Portugaly ELONP@MICROSOFT.COM
Dipankar Ray DIPANRAY@MICROSOFT.COM
Patrice Simard PATRICE@MICROSOFT.COM
Ed Snelson EDSNELSO@MICROSOFT.COM
Microsoft
1 Microsoft Way
Redmond, WA 98052, USA

Abstract

This work shows how to leverage causal inference to understand the behavior of complex learning systems interacting with their environment and predict the consequences of changes to the system. Such predictions allow both humans and algorithms to select the changes that would have improved the system performance. This work is illustrated by experiments on the ad placement system associated with the Bing search engine.

Keywords: causation, counterfactual reasoning, computational advertising

1. Introduction

Statistical machine learning technologies in the real world are never without a purpose. Using their predictions, humans or machines make decisions whose circuitous consequences often violate the modeling assumptions that justified the system design in the first place.

Such contradictions appear very clearly in the case of the learning systems that power web scale applications such as search engines, ad placement engines, or recommendation systems. For instance, the placement of advertisement on the result pages of Internet search engines depend on the bids of advertisers and on scores computed by statistical machine learning systems. Because the scores affect the contents of the result pages proposed to the users, they directly influence the occurrence of clicks and the corresponding advertiser payments. They also have important indirect effects. Ad placement decisions impact the satisfaction of the users and therefore their willingness to frequent this web site in the future. They also impact the return on investment observed by the

∗. Current address: *Jonas Peters, ETH Zürich, Rämistraße 101, 8092 Zürich, Switzerland.*
†. Current address: *Joaquin Quiñonero-Candela, Facebook, 1 Hacker Way, Menlo Park, CA 94025, USA.*

advertisers and therefore their future bids. Finally they change the nature of the data collected for training the statistical models in the future.

These complicated interactions are clarified by important theoretical works. Under simplified assumptions, mechanism design (Myerson, 1981) leads to an insightful account of the advertiser feedback loop (Varian, 2007; Edelman et al., 2007). Under simplified assumptions, multiarmed bandits theory (Robbins, 1952; Auer et al., 2002; Langford and Zhang, 2008) and reinforcement learning (Sutton and Barto, 1998) describe the exploration/exploitation dilemma associated with the training feedback loop. However, none of these approaches gives a complete account of the complex interactions found in real-life systems.

This contribution proposes a novel approach: we view these *complicated interactions as manifestations of the fundamental difference that separates correlation and causation.* Using the ad placement example as a model of our problem class, we therefore argue that *the language and the methods of causal inference* provide flexible means to *describe such complex machine learning systems* and *give sound answers to the practical questions* facing the designer of such a system. Is it useful to pass a new input signal to the statistical model? Is it worthwhile to collect and label a new training set? What about changing the loss function or the learning algorithm? In order to answer such questions and improve the operational performance of the learning system, one needs to unravel how the information produced by the statistical models traverses the web of causes and effects and eventually produces measurable performance metrics.

Readers with an interest in causal inference will find in this paper (*i*) a *real world example demonstrating the value of causal inference for large-scale machine learning applications*, (*ii*) *causal inference techniques applicable to continuously valued variables with meaningful confidence intervals*, and (*iii*) *quasi-static analysis techniques for estimating how small interventions affect certain causal equilibria*. Readers with an interest in real-life applications will find (*iv*) a selection of *practical counterfactual analysis techniques applicable to many real-life machine learning systems*. Readers with an interest in computational advertising will find a principled framework that (*v*) explains *how to soundly use machine learning techniques for ad placement*, and (*vi*) *conceptually connects machine learning and auction theory* in a compelling manner.

The paper is organized as follows. Section 2 gives an overview of the advertisement placement problem which serves as our main example. In particular, we stress some of the difficulties encountered when one approaches such a problem without a principled perspective. Section 3 provides a condensed review of the essential concepts of causal modeling and inference. Section 4 centers on formulating and answering counterfactual questions such as "how would the system have performed during the data collection period if certain interventions had been carried out on the system ?" We describe importance sampling methods for counterfactual analysis, with clear conditions of validity and confidence intervals. Section 5 illustrates how the structure of the causal graph reveals opportunities to exploit prior information and vastly improve the confidence intervals. Section 6 describes how counterfactual analysis provides essential signals that can drive learning algorithms. Assume that we have identified interventions that would have caused the system to perform well during the data collection period. Which guarantee can we obtain on the performance of these same interventions in the future? Section 7 presents counterfactual differential techniques for the study of equlibria. Using data collected when the system is at equilibrium, we can estimate how a small intervention displaces the equilibrium. This provides an elegant and effective way to reason about long-term feedback effects. Various appendices complete the main text with information that we think more relevant to readers with specific backgrounds.

2. Causation Issues in Computational Advertising

After giving an overview of the advertisement placement problem, which serves as our main example, this section illustrates some of the difficulties that arise when one does not pay sufficient attention to the causal structure of the learning system.

2.1 Advertisement Placement

All Internet users are now familiar with the advertisement messages that adorn popular web pages. Advertisements are particularly effective on search engine result pages because users who are searching for something are good targets for advertisers who have something to offer. Several actors take part in this Internet advertisement game:

- Advertisers create advertisement messages, and place bids that describe how much they are willing to pay to see their ads displayed or clicked.

- Publishers provide attractive web services, such as, for instance, an Internet search engine. They display selected ads and expect to receive payments from the advertisers. The infrastructure to collect the advertiser bids and select ads is sometimes provided by an advertising network on behalf of its affiliated publishers. For the purposes of this work, we simply consider a publisher large enough to run its own infrastructure.

- Users reveal information about their current interests, for instance, by entering a query in a search engine. They are offered web pages that contain a selection of ads (Figure 1). Users sometimes click on an advertisement and are transported to a web site controlled by the advertiser where they can initiate some business.

A conventional bidding language is necessary to precisely define under which conditions an advertiser is willing to pay the bid amount. In the case of Internet search advertisement, each bid specifies (a) the advertisement message, (b) a set of keywords, (c) one of several possible matching criteria between the keywords and the user query, and (d) the maximal price the advertiser is willing to pay when a user clicks on the ad after entering a query that matches the keywords according to the specified criterion.

Whenever a user visits a publisher web page, an advertisement placement engine runs an auction in real time in order to select winning ads, determine where to display them in the page, and compute the prices charged to advertisers, should the user click on their ad. Since the placement engine is operated by the publisher, it is designed to further the interests of the publisher. Fortunately for everyone else, the publisher must balance short term interests, namely the immediate revenue brought by the ads displayed on each web page, and long term interests, namely the future revenues resulting from the continued satisfaction of both users and advertisers.

Auction theory explains how to design a mechanism that optimizes the revenue of the seller of a single object (Myerson, 1981; Milgrom, 2004) under various assumptions about the information available to the buyers regarding the intentions of the other buyers. In the case of the ad placement problem, the publisher runs multiple auctions and sells opportunities to receive a click. When nearly identical auctions occur thousand of times per second, it is tempting to consider that the advertisers have perfect information about each other. This assumption gives support to the popular generalized second price rank-score auction (Varian, 2007; Edelman et al., 2007):

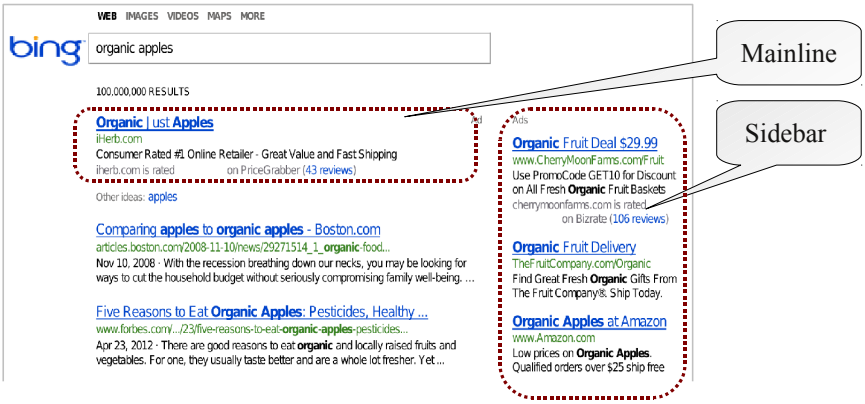


Figure 1: Mainline and sidebar ads on a search result page. Ads placed in the mainline are more likely to be noticed, increasing both the chances of a click if the ad is relevant and the risk of annoying the user if the ad is not relevant.

- Let x represent the auction context information, such as the user query, the user profile, the date, the time, etc. The ad placement engine first determines all eligible ads $a_1 \dots a_n$ and the corresponding bids $b_1 \dots b_n$ on the basis of the auction context x and of the matching criteria specified by the advertisers.

- For each selected ad a_i and each potential position p on the web page, a statistical model outputs the estimate $q_{i,p}(x)$ of the probability that ad a_i displayed in position p receives a user click. The rank-score $r_{i,p}(x) = b_i q_{i,p}(x)$ then represents the purported value associated with placing ad a_i at position p.

- Let L represent a possible ad layout, that is, a set of positions that can simultaneously be populated with ads, and let $\mathcal{L}$ be the set of possible ad layouts, including of course the empty layout. The optimal layout and the corresponding ads are obtained by maximizing the total rank-score

$$\max_{L \in \mathcal{L}} \max_{i_1, i_2, \dots} \sum_{p \in L} r_{i_p,p}(x)\,, \tag{1}$$

  subject to reserve constraints

$$\forall p \in L,\ r_{i_p,p}(x) \geq R_p(x)\,,$$

  and also subject to diverse policy constraints, such as, for instance, preventing the simultaneous display of multiple ads belonging to the same advertiser. Under mild assumptions, this discrete maximization problem is amenable to computationally efficient greedy algorithms (see appendix A.)

- The advertiser payment associated with a user click is computed using the generalized second price (GSP) rule: the advertiser pays the smallest bid that it could have entered without changing the solution of the discrete maximization problem, all other bids remaining equal. In other words, the advertiser could not have manipulated its bid and obtained the same treatment for a better price.

Under the perfect information assumption, the analysis suggests that the publisher simply needs to find which reserve prices $R_p(x)$ yield the best revenue *per auction*. However, the total revenue of the publisher also depends on the traffic experienced by its web site. Displaying an excessive number of irrelevant ads can train users to ignore the ads, and can also drive them to competing web sites. Advertisers can artificially raise the rank-scores of irrelevant ads by temporarily increasing the bids. Indelicate advertisers can create deceiving advertisements that elicit many clicks but direct users to spam web sites. Experience shows that the continued satisfaction of the users is more important to the publisher than it is to the advertisers.

Therefore the generalized second price rank-score auction has evolved. Rank-scores have been augmented with terms that quantify the user satisfaction or the ad relevance. Bids receive adaptive discounts in order to deal with situations where the perfect information assumption is unrealistic. These adjustments are driven by additional statistical models. The ad placement engine should therefore be viewed as a complex learning system interacting with both users and advertisers.

2.2 Controlled Experiments

The designer of such an ad placement engine faces the fundamental question of testing whether a proposed modification of the ad placement engine results in an improvement of the operational performance of the system.

The simplest way to answer such a question is to try the modification. The basic idea is to randomly split the users into treatment and control groups (Kohavi et al., 2008). Users from the control group see web pages generated using the unmodified system. Users of the treatment groups see web pages generated using alternate versions of the system. Monitoring various performance metrics for a couple months usually gives sufficient information to reliably decide which variant of the system delivers the most satisfactory performance.

Modifying an advertisement placement engine elicits reactions from both the users and the advertisers. Whereas it is easy to split users into treatment and control groups, splitting advertisers into treatment and control groups demands special attention because each auction involves multiple advertisers (Charles et al., 2012). Simultaneously controlling for both users and advertisers is probably impossible.

Controlled experiments also suffer from several drawbacks. They are expensive because they demand a complete implementation of the proposed modifications. They are slow because each experiment typically demands a couple months. Finally, although there are elegant ways to efficiently run overlapping controlled experiments on the same traffic (Tang et al., 2010), they are limited by the volume of traffic available for experimentation.

It is therefore difficult to rely on controlled experiments during the conception phase of potential improvements to the ad placement engine. It is similarly difficult to use controlled experiments to drive the training algorithms associated with click probability estimation models. Cheaper and faster statistical methods are needed to drive these essential aspects of the development of an ad placement engine. Unfortunately, interpreting cheap and fast data can be very deceiving.

2.3 Confounding Data

Assessing the consequence of an intervention using statistical data is generally challenging because it is often difficult to determine whether the observed effect is a simple consequence of the intervention or has other uncontrolled causes.

	Overall	Patients with small stones	Patients with large stones
Treatment A: Open surgery	78% (273/350)	**93%** (81/87)	**73%** (192/263)
Treatment B: Percutaneous nephrolithotomy	**83%** (289/350)	87% (234/270)	69% (55/80)

Table 1: A classic example of Simpson's paradox. The table reports the success rates of two treatments for kidney stones (Charig et al., 1986, Tables I and II). Although the overall success rate of treatment B seems better, treatment B performs worse than treatment A on both patients with small kidney stones and patients with large kidney stones. See Section 2.3.

For instance, the empirical comparison of certain kidney stone treatments illustrates this difficulty (Charig et al., 1986). Table 2.3 reports the success rates observed on two groups of 350 patients treated with respectively open surgery (treatment A, with 78% success) and percutaneous nephrolithotomy (treatment B, with 83% success). Although treatment B seems more successful, it was more frequently prescribed to patients suffering from small kidney stones, a less serious condition. Did treatment B achieve a high success rate because of its intrinsic qualities or because it was preferentially applied to less severe cases? Further splitting the data according to the size of the kidney stones reverses the conclusion: treatment A now achieves the best success rate for both patients suffering from large kidney stones and patients suffering from small kidney stones. Such an inversion of the conclusion is called Simpson's paradox (Simpson, 1951).

The stone size in this study is an example of a *confounding variable*, that is an uncontrolled variable whose consequences pollute the effect of the intervention. Doctors knew the size of the kidney stones, chose to treat the healthier patients with the least invasive treatment B, and therefore caused treatment B to appear more effective than it actually was. If we now decide to apply treatment B to all patients irrespective of the stone size, we break the causal path connecting the stone size to the outcome, we eliminate the illusion, and we will experience disappointing results.

When we suspect the existence of a confounding variable, we can split the contingency tables and reach improved conclusions. Unfortunately we cannot fully trust these conclusions unless we are certain to have taken into account all confounding variables. The real problem therefore comes from the confounding variables we do not know.

Randomized experiments arguably provide the only correct solution to this problem (see Stigler, 1992). The idea is to randomly chose whether the patient receives treatment A or treatment B. Because this random choice is independent from all the potential confounding variables, known and unknown, they cannot pollute the observed effect of the treatments (see also Section 4.2). This is why controlled experiments in ad placement (Section 2.2) randomly distribute users between treatment and control groups, and this is also why, in the case of an ad placement engine, we should be somehow concerned by the practical impossibility to randomly distribute both users and advertisers.

	Overall	q_2 low	q_2 high
q_1 low	6.2% (124/2000)	**5.1%** (92/1823)	**18.1%** (32/176)
q_1 high	**7.5%** (149/2000)	4.8% (71/1500)	15.6% (78/500)

Table 2: Confounding data in ad placement. The table reports the click-through rates and the click counts of the second mainline ad. The overall counts suggest that the click-through rate of the second mainline ad increases when the click probability estimate q_1 of the top ad is high. However, if we further split the pages according to the click probability estimate q_2 of the second mainline ad, we reach the opposite conclusion. See Section 2.4.

2.4 Confounding Data in Ad Placement

Let us return to the question of assessing the value of passing a new input signal to the ad placement engine click prediction model. Section 2.1 outlines a placement method where the click probability estimates $q_{i,p}(x)$ depend on the ad and the position we consider, but do not depend on other ads displayed on the page. We now consider replacing this model by a new model that additionally uses the estimated click probability of the top mainline ad to estimate the click probability of the second mainline ad (Figure 1). We would like to estimate the effect of such an intervention using existing statistical data.

We have collected ad placement data for Bing search result pages served during three consecutive hours on a certain slice of traffic. Let q_1 and q_2 denote the click probability estimates computed by the existing model for respectively the top mainline ad and the second mainline ad. After excluding pages displaying fewer than two mainline ads, we form two groups of 2000 pages randomly picked among those satisfying the conditions $q_1 < 0.15$ for the first group and $q_1 \geq 0.15$ for the second group. Table 2.4 reports the click counts and frequencies observed on the second mainline ad in each group. Although the overall numbers show that users click more often on the second mainline ad when the top mainline ad has a high click probability estimate q_1, this conclusion is reversed when we further split the data according to the click probability estimate q_2 of the second mainline ad.

Despite superficial similarities, this example is considerably more difficult to interpret than the kidney stone example. The overall click counts show that the actual click-through rate of the second mainline ad is positively correlated with the click probability estimate on the top mainline ad. Does this mean that we can increase the total number of clicks by placing regular ads below frequently clicked ads?

Remember that the click probability estimates depend on the search query which itself depends on the user intention. The most likely explanation is that pages with a high q_1 are frequently associated with more commercial searches and therefore receive more ad clicks on all positions. The observed correlation occurs because the presence of a click and the magnitude of the click probability estimate q_1 have a common cause: the user intention. Meanwhile, the click probability estimate q_2 returned by the current model for the second mainline ad also depend on the query and therefore the user intention. Therefore, assuming that this dependence has comparable strength, and assuming that there are no other causal paths, splitting the counts according to the magnitude of q_2 factors out the effects of this common confounding cause. We then observe a negative correlation which now

suggests that a frequently clicked top mainline ad has a negative impact on the click-through rate of the second mainline ad.

If this is correct, we would probably increase the accuracy of the click prediction model by switching to the new model. This would decrease the click probability estimates for ads placed in the second mainline position on commercial search pages. These ads are then less likely to clear the reserve and therefore more likely to be displayed in the less attractive sidebar. The net result is probably a loss of clicks and a loss of money despite the higher quality of the click probability model. Although we could tune the reserve prices to compensate this unfortunate effect, nothing in this data tells us where the performance of the ad placement engine will land. Furthermore, unknown confounding variables might completely reverse our conclusions.

Making sense out of such data is just too complex !

2.5 A Better Way

It should now be obvious that we need a more principled way to reason about the effect of potential interventions. We provide one such more principled approach using the causal inference machinery (Section 3). The next step is then the identification of a class of questions that are sufficiently expressive to guide the designer of a complex learning system, and sufficiently simple to be answered using data collected in the past using adequate procedures (Section 4).

A machine learning algorithm can then be viewed as an automated way to generate questions about the parameters of a statistical model, obtain the corresponding answers, and update the parameters accordingly (Section 6). Learning algorithms derived in this manner are very flexible: human designers and machine learning algorithms can cooperate seamlessly because they rely on similar sources of information.

3. Modeling Causal Systems

When we point out a causal relationship between two events, we describe what we expect to happen to the event we call the *effect*, should an external operator manipulate the event we call the *cause*. Manipulability theories of causation (von Wright, 1971; Woodward, 2005) raise this commonsense insight to the status of a definition of the causal relation. Difficult adjustments are then needed to interpret statements involving causes that we can only observe through their effects, *"because they love me,"* or that are not easily manipulated, *"because the earth is round."*

Modern statistical thinking makes a clear distinction between the statistical model and the world. The actual mechanisms underlying the data are considered unknown. The statistical models do not need to reproduce these mechanisms to emulate the observable data (Breiman, 2001). Better models are sometimes obtained by deliberately avoiding to reproduce the true mechanisms (Vapnik, 1982, Section 8.6). We can approach the manipulability puzzle in the same spirit by viewing causation as a reasoning model (Bottou, 2011) rather than a property of the world. Causes and effects are simply the pieces of an abstract reasoning game. Causal statements that are not empirically testable acquire validity when they are used as intermediate steps when one reasons about manipulations or interventions amenable to experimental validation.

This section presents the rules of this reasoning game. We largely follow the framework proposed by Pearl (2009) because it gives a clear account of the connections between causal models and probabilistic models.

$$
\begin{array}{rcll}
x & = & f_1(u,\varepsilon_1) & \text{Query context } x \text{ from user intent } u. \\
a & = & f_2(x,v,\varepsilon_2) & \text{Eligible ads } (a_i) \text{ from query } x \text{ and inventory } v. \\
b & = & f_3(x,v,\varepsilon_3) & \text{Corresponding bids } (b_i). \\
q & = & f_4(x,a,\varepsilon_4) & \text{Scores } (q_{i,p},R_p) \text{ from query } x \text{ and ads } a. \\
s & = & f_5(a,q,b,\varepsilon_5) & \text{Ad slate } s \text{ from eligible ads } a, \text{ scores } q \text{ and bids } b. \\
c & = & f_6(a,q,b,\varepsilon_6) & \text{Corresponding click prices } c. \\
y & = & f_7(s,u,\varepsilon_7) & \text{User clicks } y \text{ from ad slate } s \text{ and user intent } u. \\
z & = & f_8(y,c,\varepsilon_8) & \text{Revenue } z \text{ from clicks } y \text{ and prices } c.
\end{array}
$$

Figure 2: A structural equation model for ad placement. The sequence of equations describes the flow of information. The functions f_k describe how effects depend on their direct causes. The additional noise variables ε_k represent independent sources of randomness useful to model probabilistic dependencies.

3.1 The Flow of Information

Figure 2 gives a deterministic description of the operation of the ad placement engine. Variable u represents the user and his or her intention in an unspecified manner. The query and query context x is then expressed as an unknown function of the u and of a noise variable ε_1. Noise variables in this framework are best viewed as independent sources of randomness useful for modeling a nondeterministic causal dependency. We shall only mention them when they play a specific role in the discussion. The set of eligible ads a and the corresponding bids b are then derived from the query x and the ad inventory v supplied by the advertisers. Statistical models then compute a collection of scores q such as the click probability estimates $q_{i,p}$ and the reserves R_p introduced in Section 2.1. The placement logic uses these scores to generate the "ad slate" s, that is, the set of winning ads and their assigned positions. The corresponding click prices c are computed. The set of user clicks y is expressed as an unknown function of the ad slate s and the user intent u. Finally the revenue z is expressed as another function of the clicks y and the prices c.

Such a system of equations is named *structural equation model* (Wright, 1921). Each equation asserts a functional dependency between an effect, appearing on the left hand side of the equation, and its direct causes, appearing on the right hand side as arguments of the function. Some of these causal dependencies are *unknown*. Although we postulate that the effect can be expressed as some function of its direct causes, we do not know the form of this function. For instance, the designer of the ad placement engine knows functions f_2 to f_6 and f_8 because he has designed them. However, he does not know the functions f_1 and f_7 because whoever designed the user did not leave sufficient documentation.

Figure 3 represents the directed causal graph associated with the structural equation model. Each arrow connects a direct cause to its effect. The noise variables are omitted for simplicity. The structure of this graph reveals fundamental assumptions about our model. For instance, the user clicks y do not directly depend on the scores q or the prices c because users do not have access to this information.

We hold as a principle that causation obeys the arrow of time: causes always precede their effects. Therefore the causal graph must be *acyclic*. Structural equation models then support two fundamental operations, namely simulation and intervention.

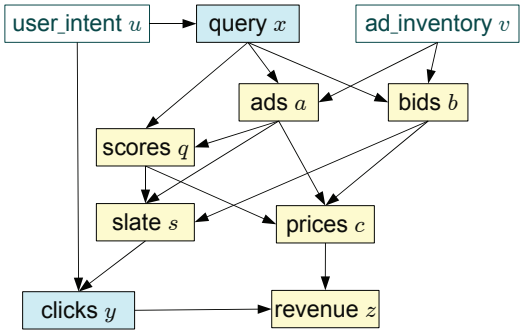


Figure 3: Causal graph associated with the structural equation model of Figure 2. The mutually independent noise variables ε_1 to ε_8 are implicit. The variables a, b, q, s, c, and z depend on their direct causes in known ways. In contrast, the variables u and v are exogenous and the variables x and y depend on their direct causes through unknown functions.

- *Simulation* – Let us assume that we know both the exact form of all functional dependencies and the value of all exogenous variables, that is, the variables that never appear in the left hand side of an equation. We can compute the values of all the remaining variables by applying the equations in their natural time sequence.

- *Intervention* – As long as the causal graph remains acyclic, we can construct derived structural equation models using arbitrary algebraic manipulations of the system of equations. For instance, we can clamp a variable to a constant value by rewriting the right-hand side of the corresponding equation as the specified constant value.

The algebraic manipulation of the structural equation models provides a powerful language to describe interventions on a causal system. This is not a coincidence. Many aspects of the mathematical notation were invented to support causal inference in classical mechanics. However, we no longer have to interpret the variable values as physical quantities: the equations simply describe the flow of information in the causal model (Wiener, 1948).

3.2 The Isolation Assumption

Let us now turn our attention to the exogenous variables, that is, variables that never appear in the left hand side of an equation of the structural model. Leibniz's *principle of sufficient reason* claims that there are no facts without causes. This suggests that the exogenous variables are the effects of a network of causes not expressed by the structural equation model. For instance, the user intent u and the ad inventory v in Figure 3 have temporal correlations because both users and advertisers worry about their budgets when the end of the month approaches. Any structural equation model should then be understood in the context of a larger structural equation model potentially describing all things in existence.

Ads served on a particular page contribute to the continued satisfaction of both users and advertisers, and therefore have an effect on their willingness to use the services of the publisher in the future. The ad placement structural equation model shown in Figure 2 only describes the causal dependencies for a single page and therefore cannot account for such effects. Consider however a very

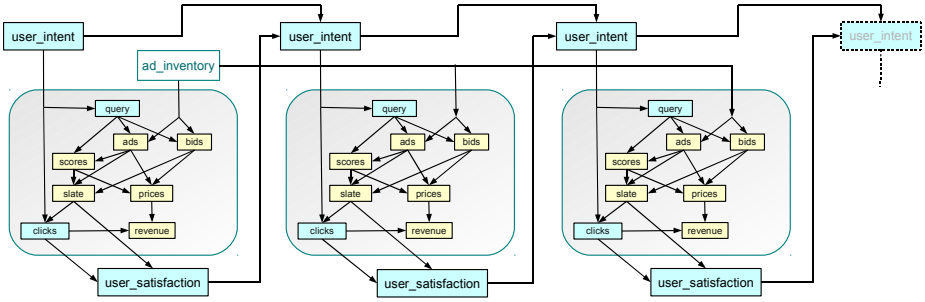


Figure 4: Conceptually unrolling the user feedback loop by threading instances of the single page causal graph (Figure 3). Both the ad slate s_t and user clicks y_t have an indirect effect on the user intent u_{t+1} associated with the next query.

large structural equation model containing a copy of the page-level model for every web page ever served by the publisher. Figure 4 shows how we can thread the page-level models corresponding to pages served to the same user. Similarly we could model how advertisers track the performance and the cost of their advertisements and model how their satisfaction affects their future bids. The resulting causal graphs can be very complex. Part of this complexity results from time-scale differences. Thousands of search pages are served in a second. Each page contributes a little to the continued satisfaction of one user and a few advertisers. The accumulation of these contributions produces measurable effects after a few weeks.

Many of the functional dependencies expressed by the structural equation model are left unspecified. Without direct knowledge of these functions, we must reason using statistical data. The most fundamental statistical data is collected from repeated trials that are assumed independent. When we consider the large structured equation model of everything, we can only have one large trial producing a single data point.[1] It is therefore desirable to identify repeated patterns of identical equations that can be viewed as repeated independent trials. Therefore, when we study a structural equation model representing such a pattern, we need to make an additional assumption to expresses the idea that the outcome of one trial does not affect the other trials. We call such an assumption an *isolation assumption* by analogy with thermodynamics.[2] This can be achieved by assuming that *the exogenous variables are independently drawn from an unknown but fixed joint probability distribution*. This assumption cuts the causation effects that could flow through the exogenous variables.

The noise variables are also exogenous variables acting as independent source of randomness. The noise variables are useful to represent the conditional distribution $\mathrm{P}(\,\mathsf{effect}\,|\,\mathsf{causes}\,)$ using the equation $\mathsf{effect} = f(\mathsf{causes}, \varepsilon)$. Therefore, we also assume joint independence between all the noise variables and any of the named exogenous variable.[3] For instance, in the case of the ad placement

1. See also the discussion on reinforcement learning, Section 3.5.
2. The concept of isolation is pervasive in physics. An isolated system in thermodynamics (Reichl, 1998, Section 2.D) or a closed system in mechanics (Landau and Lifshitz, 1969, §5) evolves without exchanging mass or energy with its surroundings. Experimental trials involving systems that are assumed isolated may differ in their initial setup and therefore have different outcomes. Assuming isolation implies that the outcome of each trial cannot affect the other trials.
3. Rather than letting two noise variables display measurable statistical dependencies because they share a common cause, we prefer to name the common cause and make the dependency explicit in the graph.

$$\mathrm{P}\left(\begin{array}{c} u,v,x,a,b \\ q,s,c,y,z \end{array}\right) = \left\{\begin{array}{ll} \mathrm{P}(u,v) & \text{Exogenous vars.} \\ \times\, \mathrm{P}(x\,|\,u) & \text{Query.} \\ \times\, \mathrm{P}(a\,|\,x,v) & \text{Eligible ads.} \\ \times\, \mathrm{P}(b\,|\,x,v) & \text{Bids.} \\ \times\, \mathrm{P}(q\,|\,x,a) & \text{Scores.} \\ \times\, \mathrm{P}(s\,|\,a,q,b) & \text{Ad slate.} \\ \times\, \mathrm{P}(c\,|\,a,q,b) & \text{Prices.} \\ \times\, \mathrm{P}(y\,|\,s,u) & \text{Clicks.} \\ \times\, \mathrm{P}(z\,|\,y,c) & \text{Revenue.} \end{array}\right.$$

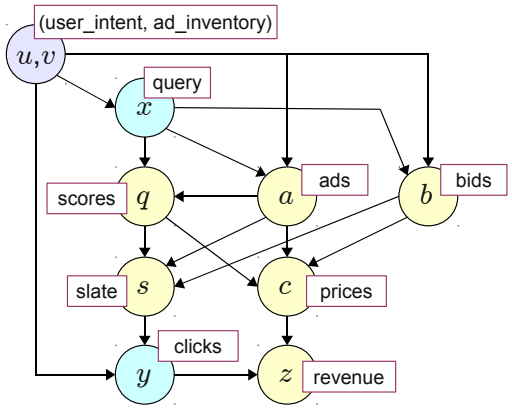


Figure 6: Bayesian network associated with the Markov factorization shown in Figure 5.

model shown in Figure 2, we assume that the joint distribution of the exogenous variables factorizes as

$$\mathrm{P}(u,v,\varepsilon_1,\ldots,\varepsilon_8) = \mathrm{P}(u,v)\,\mathrm{P}(\varepsilon_1)\ldots\mathrm{P}(\varepsilon_8)\,.$$

Since an isolation assumption is only true up to a point, it should be expressed clearly and remain under constant scrutiny. We must therefore measure additional performance metrics that reveal how the isolation assumption holds. For instance, the ad placement structural equation model and the corresponding causal graph (figures 2 and 3) do not take user feedback or advertiser feedback into account. Measuring the revenue is not enough because we could easily generate revenue at the expense of the satisfaction of the users and advertisers. When we evaluate interventions under such an isolation assumption, we also need to measure a battery of additional quantities that act as proxies for the user and advertiser satisfaction. Noteworthy examples include ad relevance estimated by human judges, and advertiser surplus estimated from the auctions (Varian, 2009).

3.3 Markov Factorization

Conceptually, we can draw a sample of the exogenous variables using the distribution specified by the isolation assumption, and we can then generate values for all the remaining variables by simulating the structural equation model.

This process defines a *generative probabilistic model* representing the joint distribution of all variables in the structural equation model. The distribution readily factorizes as the product of the joint probability of the named exogenous variables, and, for each equation in the structural equation model, the conditional probability of the effect given its direct causes (Spirtes et al., 1993; Pearl, 2000). As illustrated by figures 5 and 6, this *Markov factorization* connects the structural equation model that describes causation, and the Bayesian network that describes the joint probability distribution followed by the variables under the isolation assumption.[4]

Structural equation models and Bayesian networks appear so intimately connected that it could be easy to forget the differences. The structural equation model is an algebraic object. As long as the causal graph remains acyclic, algebraic manipulations are interpreted as interventions on the causal system. The Bayesian network is a generative statistical model representing a class of joint probability distributions, and, as such, does not support algebraic manipulations. However, the symbolic representation of its Markov factorization is an algebraic object, essentially equivalent to the structural equation model.

3.4 Identification, Transportation, and Transfer Learning

Consider a causal system represented by a structural equation model with some unknown functional dependencies. Subject to the isolation assumption, data collected during the operation of this system follows the distribution described by the corresponding Markov factorization. Let us first assume that this data is sufficient to identify the joint distribution of the subset of variables we can observe. We can intervene on the system by clamping the value of some variables. This amounts to replacing the right-hand side of the corresponding structural equations by constants. The joint distribution of the variables is then described by a new Markov factorization that shares many factors with the original Markov factorization. Which conditional probabilities associated with this new distribution can we express using only conditional probabilities identified during the observation of the original system? This is called the *identifiability* problem. More generally, we can consider arbitrarily complex manipulations of the structural equation model, and we can perform multiple experiments involving different manipulations of the causal system. Which conditional probabilities pertaining to one experiment can be expressed using only conditional probabilities identified during the observation of other experiments? This is called the *transportability* problem.

Pearl's *do*-calculus completely solves the identifiability problem and provides useful tools to address many instances of the transportability problem (see Pearl, 2012). Assuming that we *know* the conditional probability distributions involving observed variables in the original structural equation model, *do*-calculus allows us to *derive* conditional distributions pertaining to the manipulated structural equation model.

Unfortunately, we must further distinguish the conditional probabilities that we know (because we designed them) from those that we estimate from empirical data. This distinction is important because estimating the distribution of continuous or high cardinality variables is notoriously difficult. Furthermore, *do*-calculus often combines the estimated probabilities in ways that amplify estimation errors. This happens when the manipulated structural equation model exercises the variables in ways that were rarely observed in the data collected from the original structural equation model.

4. Bayesian networks are directed graphs representing the Markov factorization of a joint probability distribution: the arrows no longer have a causal interpretation.

Therefore we prefer to use much simpler causal inference techniques (see sections 4.1 and 4.2). Although these techniques do not have the completeness properties of *do*-calculus, they combine estimation and transportation in a manner that facilitates the derivation of useful confidence intervals.

3.5 Special Cases

Three special cases of causal models are particularly relevant to this work.

- In the multi-armed bandit (Robbins, 1952), a user-defined policy function π determines the distribution of action $a \in \{1 \dots K\}$, and an unknown reward function r determines the distribution of the outcome y given the action a (Figure 7). In order to maximize the accumulated rewards, the player must construct policies π that balance the exploration of the action space with the exploitation of the best action identified so far (Auer et al., 2002; Audibert et al., 2007; Seldin et al., 2012).

- The contextual bandit problem (Langford and Zhang, 2008) significantly increases the complexity of multi-armed bandits by adding one exogenous variable x to the policy function π and the reward functions r (Figure 8).

- Both multi-armed bandit and contextual bandit are special case of reinforcement learning (Sutton and Barto, 1998). In essence, a Markov decision process is a sequence of contextual bandits where the context is no longer an exogenous variable but a state variable that depends on the previous states and actions (Figure 9). Note that the policy function π, the reward function r, and the transition function s are independent of time. All the time dependencies are expressed using the states s_t.

These special cases have increasing generality. Many simple structural equation models can be reduced to a contextual bandit problem using appropriate definitions of the context x, the action a and the outcome y. For instance, assuming that the prices c are discrete, the ad placement structural equation model shown in Figure 2 reduces to a contextual bandit problem with context (u,v), actions (s,c) and reward z. Similarly, given a sufficiently intricate definition of the state variables s_t, all structural equation models with discrete variables can be reduced to a reinforcement learning problem. Such reductions lose the fine structure of the causal graph. We show in Section 5 how this fine structure can in fact be leveraged to obtain more information from the same experiments.

Modern reinforcement learning algorithms (see Sutton and Barto, 1998) leverage the assumption that the policy function, the reward function, the transition function, and the distributions of the corresponding noise variables, are independent from time. This invariance property provides great benefits when the observed sequences of actions and rewards are long in comparison with the size of the state space. Only Section 7 in this contribution presents methods that take advantage of such an invariance. The general question of leveraging arbitrary functional invariances in causal graphs is left for future work.

4. Counterfactual Analysis

We now return to the problem of formulating and answering questions about the value of proposed changes of a learning system. Assume for instance that we consider replacing the score computation

$$\begin{array}{rcll} a & = & \pi(\varepsilon) & \text{Action } a \in \{1 \dots K\} \\ y & = & r(a, \varepsilon') & \text{Reward } y \in \mathbb{R} \end{array}$$

Figure 7: Structural equation model for the multi-armed bandit problem. The policy π selects a discrete action a, and the reward function r determines the outcome y. The noise variables ε and ε' represent independent sources of randomness useful to model probabilistic dependencies.

$$\begin{array}{rcll} a & = & \pi(x, \varepsilon) & \text{Action } a \in \{1 \dots K\} \\ y & = & r(x, a, \varepsilon') & \text{Reward } y \in \mathbb{R} \end{array}$$

Figure 8: Structural equation model for contextual bandit problem. Both the action and the reward depend on an exogenous context variable x.

$$\begin{array}{rcll} a_t & = & \pi(s_{t-1}, \varepsilon_t) & \text{Action} \\ y_t & = & r(s_{t-1}, a_t, \varepsilon'_t) & \text{Reward } r_t \in \mathbb{R} \\ s_t & = & s(s_{t-1}, a_t, \varepsilon''_t) & \text{Next state} \end{array}$$

Figure 9: Structural equation model for reinforcement learning. The above equations are replicated for all $t \in \{0 \dots, T\}$. The context is now provided by a state variable s_{t-1} that depends on the previous states and actions.

model M of an ad placement engine by an alternate model M^*. We seek an answer to the conditional question:

"*How will the system perform if we replace model M by model M^* ?*"

Given sufficient time and sufficient resources, we can obtain the answer using a controlled experiment (Section 2.2). However, instead of carrying out a new experiment, we would like to obtain an answer using data that we have already collected in the past.

"*How would the system have performed if, when the data was collected, we had replaced model M by model M^*?*"

The answer of this *counterfactual question* is of course a *counterfactual statement* that describes the system performance subject to a condition that did not happen.

Counterfactual statements challenge ordinary logic because they depend on a condition that is known to be false. Although assertion $A \Rightarrow B$ is always true when assertion A is false, we certainly do not mean for all counterfactual statements to be true. Lewis (1973) navigates this paradox using a modal logic in which a counterfactual statement describes the state of affairs in an alternate world that resembles ours except for the specified differences. Counterfactuals indeed offer many subtle ways to qualify such alternate worlds. For instance, we can easily describe isolation assumptions (Section 3.2) in a counterfactual question:

"*How would the system have performed if, when the data was collected, we had replaced model M by model M^* without incurring user or advertiser reactions?*"

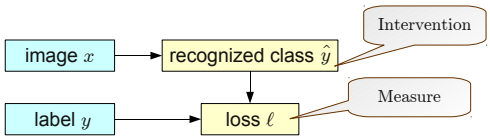


Figure 10: Causal graph for an image recognition system. We can estimate counterfactuals by replaying data collected in the past.

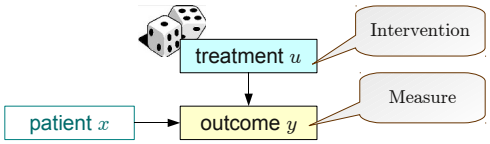


Figure 11: Causal graph for a randomized experiment. We can estimate certain counterfactuals by reweighting data collected in the past.

The fact that we could not have changed the model without incurring the user and advertiser reactions does not matter any more than the fact that we did not replace model M by model M^* in the first place. This does not prevent us from using counterfactual statements to reason about causes and effects. Counterfactual questions and statements provide a natural framework to express and share our conclusions.

The remaining text in this section explains how we can answer certain counterfactual questions using data collected in the past. More precisely, we seek to estimate performance metrics that can be expressed as expectations with respect to the distribution that would have been observed if the counterfactual conditions had been in force.[5]

4.1 Replaying Empirical Data

Figure 10 shows the causal graph associated with a simple image recognition system. The classifier takes an image x and produces a prospective class label $\hat{y}$. The loss measures the penalty associated with recognizing class $\hat{y}$ while the true class is y.

To estimate the expected error of such a classifier, we collect a representative data set composed of labelled images, run the classifier on each image, and average the resulting losses. In other words, we *replay* the data set to estimate what (counterfactual) performance would have been observed if we had used a different classifier. We can then select in retrospect the classifier that would have worked the best and hope that it will keep working well. This is the counterfactual viewpoint on empirical risk minimization (Vapnik, 1982).

Replaying the data set works because both the alternate classifier and the loss function are known. More generally, to estimate a counterfactual by replaying a data set, we need to know all the functional dependencies associated with all causal paths connecting the intervention point to the measurement point. This is obviously not always the case.

5. Although counterfactual expectations can be viewed as expectations of unit-level counterfactuals (Pearl, 2009, Definition 4), they elude the semantic subtleties of unit-level counterfactuals and can be measured with randomized experiments (see Section 4.2.)

4.2 Reweighting Randomized Trials

Figure 11 illustrates the randomized experiment suggested in Section 2.3. The patients are randomly split into two equally sized groups receiving respectively treatments A and B. The overall success rate for this experiment is therefore $Y = (Y_A + Y_B)/2$ where Y_A and Y_B are the success rates observed for each group. We would like to estimate which (counterfactual) overall success rate Y^* would have been observed if we had selected treatment A with probability p and treatment B with probability $1-p$.

Since we do not know how the outcome depends on the treatment and the patient condition, we cannot compute which outcome y^* would have been obtained if we had treated patient x with a different treatment u^*. Therefore we cannot answer this question by replaying the data as we did in Section 4.1.

However, observing different success rates Y_A and Y_B for the treatment groups reveals an empirical correlation between the treatment u and the outcome y. Since the only cause of the treatment u is an independent roll of the dices, this correlation cannot result from any known or unknown confounding common cause.[6] Having eliminated this possibility, we can *reweight* the observed outcomes and compute the estimate $Y^* \approx pY_A + (1-p)Y_B$.

4.3 Markov Factor Replacement

The reweighting approach can in fact be applied under much less stringent conditions. Let us return to the ad placement problem to illustrate this point.

The average number of ad clicks per page is often called *click yield*. Increasing the click yield usually benefits both the advertiser and the publisher, whereas increasing the revenue per page often benefits the publisher at the expense of the advertiser. Click yield is therefore a very useful metric when we reason with an isolation assumption that ignores the advertiser reactions to pricing changes.

Let ω be a shorthand for all variables appearing in the Markov factorization of the ad placement structural equation model,

$$\begin{aligned} \mathrm{P}(\omega) \;=\; & \mathrm{P}(u,v)\,\mathrm{P}(x\,|\,u)\,\mathrm{P}(a\,|\,x,v)\,\mathrm{P}(b\,|\,x,v)\,\mathrm{P}(q\,|\,x,a) \\ & \times \mathrm{P}(s\,|\,a,q,b)\,\mathrm{P}(c\,|\,a,q,b)\,\mathrm{P}(y\,|\,s,u)\,\mathrm{P}(z\,|\,y,c)\ . \end{aligned} \tag{2}$$

Variable y was defined in Section 3.1 as the set of user clicks. In the rest of the document, we slightly abuse this notation by using the same letter y to represent the number of clicks. We also write the expectation $Y = \mathbb{E}_{\omega\sim\mathrm{P}(\omega)}[y]$ using the integral notation

$$Y = \int_\omega y\,\mathrm{P}(\omega)\ .$$

We would like to estimate what the expected click yield Y^* would have been if we had used a different scoring function (Figure 12). This intervention amounts to replacing the actual factor $\mathrm{P}(q\,|\,x,a)$ by a counterfactual factor $\mathrm{P}^*(q\,|\,x,a)$ in the Markov factorization.

$$\begin{aligned} \mathrm{P}^*(\omega) \;=\; & \mathrm{P}(u,v)\,\mathrm{P}(x\,|\,u)\,\mathrm{P}(a\,|\,x,v)\,\mathrm{P}(b\,|\,x,v)\,\mathrm{P}^*(q\,|\,x,a) \\ & \times \mathrm{P}(s\,|\,a,q,b)\,\mathrm{P}(c\,|\,a,q,b)\,\mathrm{P}(y\,|\,s,u)\,\mathrm{P}(z\,|\,x,c)\ . \end{aligned} \tag{3}$$

6. See also the discussion of Reichenbach's common cause principle and of its limitations in Spirtes et al. (1993) and Spirtes and Scheines (2004).

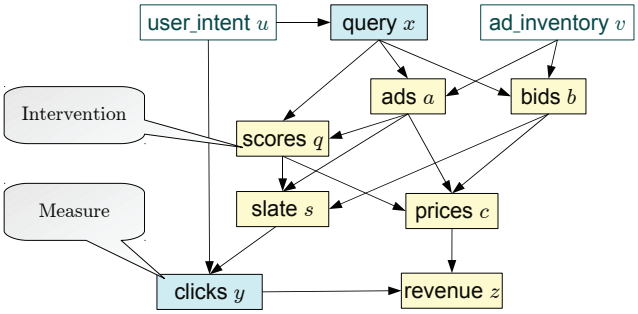


Figure 12: Estimating which average number of clicks per page would have been observed if we had used a different scoring model.

Let us assume, for simplicity, that the actual factor $\mathrm{P}(q\,|\,x,a)$ is nonzero everywhere. We can then estimate the counterfactual expected click yield Y^* using the transformation

$$Y^* \;=\; \int_\omega y\,\mathrm{P}^*(\omega) \;=\; \int_\omega y\,\frac{\mathrm{P}^*(q\,|\,x,a)}{\mathrm{P}(q\,|\,x,a)}\,\mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^{n} y_i\,\frac{\mathrm{P}^*(q_i\,|\,x_i,a_i)}{\mathrm{P}(q_i\,|\,x_i,a_i)}\,, \tag{4}$$

where the data set of tuples (a_i,x_i,q_i,y_i) is distributed according to the actual Markov factorization instead of the counterfactual Markov factorization. This data could therefore have been collected during the normal operation of the ad placement system. Each sample is reweighted to reflect its probability of occurrence under the counterfactual conditions.

In general, we can use *importance sampling* to estimate the counterfactual expectation of any quantity $\ell(\omega)$:

$$Y^* \;=\; \int_\omega \ell(\omega)\,\mathrm{P}^*(\omega) \;=\; \int_\omega \ell(\omega)\,\frac{\mathrm{P}^*(\omega)}{\mathrm{P}(\omega)}\,\mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^{n} \ell(\omega_i)\,w_i \tag{5}$$

with weights

$$w_i \;=\; w(\omega_i) \;=\; \frac{\mathrm{P}^*(\omega_i)}{\mathrm{P}(\omega_i)} \;=\; \frac{\text{factors appearing in } \mathrm{P}^*(\omega_i) \text{ but not in } \mathrm{P}(\omega_i)}{\text{factors appearing in } \mathrm{P}(\omega_i) \text{ but not in } \mathrm{P}^*(\omega_i)}\,. \tag{6}$$

Equation (6) emphasizes the simplifications resulting from the algebraic similarities of the actual and counterfactual Markov factorizations. Because of these simplifications, the evaluation of the weights only requires the knowledge of the few factors that differ between $\mathrm{P}(\omega)$ and $\mathrm{P}^*(\omega)$. Each data sample needs to provide the value of $\ell(\omega_i)$ and the values of all variables needed to evaluate the factors that do not cancel in the ratio (6).

In contrast, the replaying approach (Section 4.1) demands the knowledge of all factors of $\mathrm{P}^*(\omega)$ connecting the point of intervention to the point of measurement $\ell(\omega)$. On the other hand, it does not require the knowledge of factors appearing only in $\mathrm{P}(\omega)$.

Importance sampling relies on the assumption that all the factors appearing in the denominator of the reweighting ratio (6) are nonzero whenever the factors appearing in the numerator are nonzero. Since these factors represents conditional probabilities resulting from the effect of an independent noise variable in the structural equation model, this assumption means that the data

must be collected with an experiment involving active randomization. We must therefore design cost-effective randomized experiments that yield enough information to estimate many interesting counterfactual expectations with sufficient accuracy. This problem cannot be solved without answering the confidence interval question: given data collected with a certain level of randomization, with which accuracy can we estimate a given counterfactual expectation?

4.4 Confidence Intervals

At first sight, we can invoke the law of large numbers and write

$$Y^* = \int_\omega \ell(\omega)\, w(\omega)\, \mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^{n} \ell(\omega_i)\, w_i\,. \tag{7}$$

For sufficiently large n, the central limit theorem provides confidence intervals whose width grows with the standard deviation of the product $\ell(\omega)\, w(\omega)$.

Unfortunately, when $\mathrm{P}(\omega)$ is small, the reweighting ratio $w(\omega)$ takes large values with low probability. This heavy tailed distribution has annoying consequences because the variance of the integrand could be very high or infinite. When the variance is infinite, the central limit theorem does not hold. When the variance is merely very large, the central limit convergence might occur too slowly to justify such confidence intervals. Importance sampling works best when the actual distribution and the counterfactual distribution overlap.

When the counterfactual distribution has significant mass in domains where the actual distribution is small, the few samples available in these domains receive very high weights. Their noisy contribution dominates the reweighted estimate (7). We can obtain better confidence intervals by eliminating these few samples drawn in poorly explored domains. The resulting bias can be bounded using prior knowledge, for instance with an assumption about the range of values taken by $\ell(\omega)$,

$$\forall \omega \quad \ell(\omega) \;\in\; [0, M]\,. \tag{8}$$

Let us choose the maximum weight value R deemed acceptable for the weights. We have obtained very consistent results in practice with R equal to the fifth largest reweighting ratio observed on the empirical data.[7] We can then rely on *clipped weights* to eliminate the contribution of the poorly explored domains,

$$\bar{w}(\omega) \;=\; \begin{cases} w(\omega) & \text{if } \mathrm{P}^*(\omega) < R\,\mathrm{P}(\omega) \\ 0 & \text{otherwise.} \end{cases}$$

The condition $\mathrm{P}^*(\omega) < R\,\mathrm{P}(\omega)$ ensures that the ratio has a nonzero denominator $\mathrm{P}(\omega)$ and is smaller than R. Let Ω_R be the set of all values of ω associated with acceptable ratios:

$$\Omega_R \;=\; \{\, \omega :\; \mathrm{P}^*(\omega) < R\,\mathrm{P}(\omega) \,\}\,.$$

We can decompose Y^* in two terms:

$$Y^* \;=\; \int_{\omega\in\Omega_R} \ell(\omega)\,\mathrm{P}^*(\omega) \;+\; \int_{\omega\in\Omega\setminus\Omega_R} \ell(\omega)\,\mathrm{P}^*(\omega) \;=\; \bar{Y}^* + (Y^* - \bar{Y}^*)\,. \tag{9}$$

7. This is in fact a slight abuse because the theory calls for choosing R before seeing the data.

The first term of this decomposition is the *clipped expectation* $\bar{Y}^*$. Estimating the clipped expectation $\bar{Y}^*$ is much easier than estimating Y^* from (7) because the clipped weights $\bar{w}(\omega)$ are bounded by R.

$$\bar{Y}^* \;=\; \int_{\omega\in\Omega_R} \ell(\omega)\,\mathrm{P}^*(\omega) \;=\; \int_{\omega} \ell(\omega)\,\bar{w}(\omega)\,\mathrm{P}(\omega) \;\approx\; \widehat{Y}^* \;=\; \frac{1}{n}\sum_{i=1}^{n} \ell(\omega_i)\,\bar{w}(\omega_i)\,. \tag{10}$$

The second term of Equation (9) can be bounded by leveraging assumption (8). The resulting bound can then be conveniently estimated using only the clipped weights.

$$Y^*-\bar{Y}^* \;=\; \int_{\omega\in\Omega\setminus\Omega_R} \ell(\omega)\,\mathrm{P}^*(\omega) \;\in\; \Big[0,\, M\,\mathrm{P}^*(\Omega\setminus\Omega_R)\Big] \;=\; \Big[0,\, M\,(1-\bar{W}^*)\Big] \quad \text{with}$$

$$\bar{W}^* \;=\; \mathrm{P}^*(\Omega_R) \;=\; \int_{\omega\in\Omega_R} \mathrm{P}^*(\omega) \;=\; \int_{\omega} \bar{w}(\omega)\,\mathrm{P}(\omega) \;\approx\; \widehat{W}^* \;=\; \frac{1}{n}\sum_{i=1}^{n} \bar{w}(\omega_i)\,. \tag{11}$$

Since the clipped weights are bounded, the estimation errors associated with (10) and (11) are well characterized using either the central limit theorem or using empirical Bernstein bounds (see appendix B for details). Therefore we can derive an *outer confidence interval* of the form

$$\mathbb{P}\Big\{\; \widehat{Y}^*-\varepsilon_R \;\le\; \bar{Y}^* \;\le\; \widehat{Y}^*+\varepsilon_R \;\Big\} \;\ge\; 1-\delta \tag{12}$$

and an *inner confidence interval* of the form

$$\mathbb{P}\Big\{\; \bar{Y}^* \;\le\; Y^* \;\le\; \bar{Y}^*+M(1-\widehat{W}^*+\xi_R) \;\Big\} \;\ge\; 1-\delta\,. \tag{13}$$

The names *inner* and *outer* are in fact related to our preferred way to visualize these intervals (e.g., Figure 13). Since the bounds on $Y^*-\bar{Y}^*$ can be written as

$$\bar{Y}^* \;\le\; Y^* \;\le\; \bar{Y}^*+M\,(1-\bar{W}^*)\,, \tag{14}$$

we can derive our final confidence interval,

$$\mathbb{P}\Big\{\; \widehat{Y}^*-\varepsilon_R \;\le\; Y^* \;\le\; \widehat{Y}^*+M(1-\widehat{W}^*+\xi_R)+\varepsilon_R \;\Big\} \ge 1-2\delta\,. \tag{15}$$

In conclusion, replacing the unbiased importance sampling estimator (7) by the clipped importance sampling estimator (10) with a suitable choice of R leads to improved confidence intervals. Furthermore, since the derivation of these confidence intervals does not rely on the assumption that $\mathrm{P}(\omega)$ is nonzero everywhere, the clipped importance sampling estimator remains valid when the distribution $\mathrm{P}(\omega)$ has a limited support. This relaxes the main restriction associated with importance sampling.

4.5 Interpreting the Confidence Intervals

The estimation of the counterfactual expectation Y^* can be inaccurate because the sample size is insufficient or because the sampling distribution $\mathrm{P}(\omega)$ does not sufficiently explore the counterfactual conditions of interest.

By construction, the clipped expectation $\bar{Y}^*$ ignores the domains poorly explored by the sampling distribution $\mathrm{P}(\omega)$. The difference $Y^*-\bar{Y}^*$ then reflects the inaccuracy resulting from a lack of exploration. Therefore, assuming that the bound R has been chosen competently, the relative sizes of the outer and inner confidence intervals provide precious cues to determine whether we can continue collecting data using the same experimental setup or should adjust the data collection experiment in order to obtain a better coverage.

- The *inner confidence interval* (13) witnesses the uncertainty associated with the domain G_R insufficiently explored by the actual distribution. A large inner confidence interval suggests that the most practical way to improve the estimate is to adjust the data collection experiment in order to obtain a better coverage of the counterfactual conditions of interest.

- The *outer confidence interval* (12) represents the uncertainty that results from the limited sample size. A large outer confidence interval indicates that the sample is too small. To improve the result, we simply need to continue collecting data using the same experimental setup.

4.6 Experimenting with Mainline Reserves

We return to the ad placement problem to illustrate the reweighting approach and the interpretation of the confidence intervals. Manipulating the reserves $R_p(x)$ associated with the mainline positions (Figure 1) controls which ads are prominently displayed in the mainline or displaced into the sidebar.

We seek in this section to answer counterfactual questions of the form:

> "*How would the ad placement system have performed if we had scaled the mainline reserves by a constant factor* ρ*, without incurring user or advertiser reactions?*"

Randomization was introduced using a modified version of the ad placement engine. Before determining the ad layout (see Section 2.1), a random number ε is drawn according to the standard normal distribution $\mathcal{N}(0,1)$, and all the mainline reserves are multiplied by $m = \rho\, e^{-\sigma^2/2+\sigma\varepsilon}$. Such multipliers follow a log-normal distribution[8] whose mean is ρ and whose width is controlled by σ. This effectively provides a parametrization of the conditional score distribution $\mathrm{P}(q\,|\,x,a)$ (see Figure 5.)

The Bing search platform offers many ways to select traffic for controlled experiments (Section 2.2). In order to match our isolation assumption, individual page views were randomly assigned to traffic buckets without regard to the user identity. The main treatment bucket was processed with mainline reserves randomized by a multiplier drawn as explained above with $\rho = 1$ and $\sigma = 0.3$. With these parameters, the mean multiplier is exactly 1, and 95% of the multipliers are in range $[0.52, 1.74]$. Samples describing 22 million search result pages were collected during five consecutive weeks.

We then use this data to estimate what would have been measured if the mainline reserve multipliers had been drawn according to a distribution determined by parameters ρ^* and σ^*. This is achieved by reweighting each sample ω_i with

$$w_i = \frac{\mathrm{P}^*(q_i\,|\,x_i,a_i)}{\mathrm{P}(q_i\,|\,x_i,a_i)} = \frac{p(m_i\,;\rho^*,\sigma^*)}{p(m_i\,;\rho,\sigma)}\,,$$

where m_i is the multiplier drawn for this sample during the data collection experiment, and $p(t\,;\rho,\sigma)$ is the density of the log-normal multiplier distribution.

Figure 13 reports results obtained by varying ρ^* while keeping $\sigma^* = \sigma$. This amounts to estimating what would have been measured if all mainline reserves had been multiplied by ρ^* while keeping the same randomization. The curves bound 95% confidence intervals on the variations of the average number of mainline ads displayed per page, the average number of ad clicks per page,

8. More precisely, $ln\mathcal{N}(\mu,\sigma^2)$ with $\mu = \sigma^2/2 + \log\rho$.

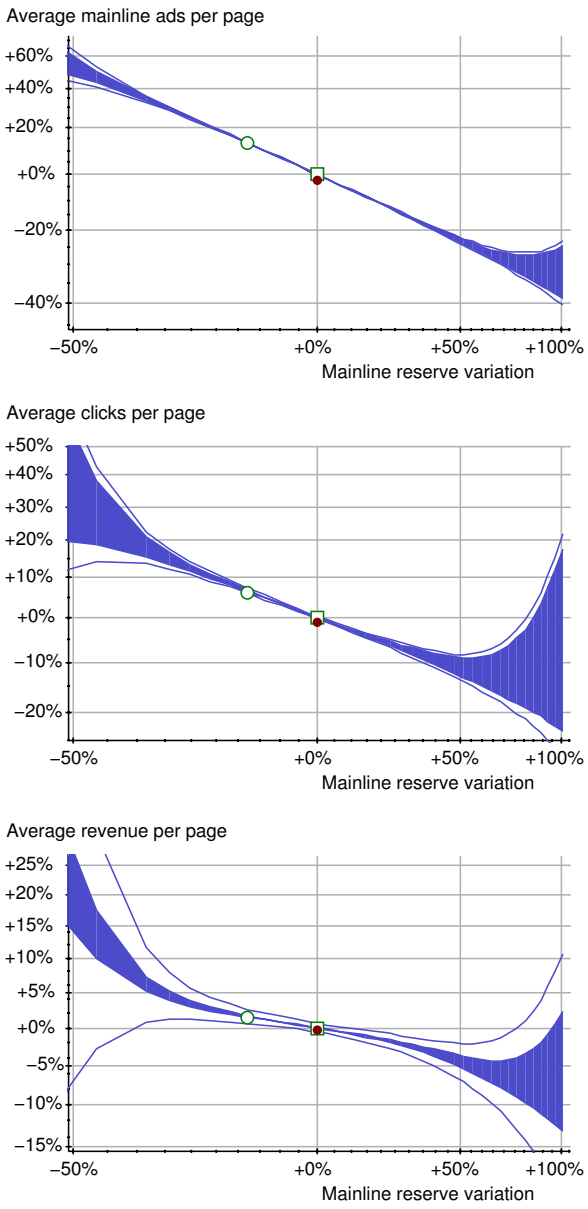


Figure 13: Estimated variations of three performance metrics in response to mainline reserve changes. The curves delimit 95% confidence intervals for the metrics we would have observed if we had increased the mainline reserves by the percentage shown on the horizontal axis. The filled areas represent the inner confidence intervals. The hollow squares represent the metrics measured on the experimental data. The hollow circles represent metrics measured on a second experimental bucket with mainline reserves reduced by 18%. The filled circles represent the metrics effectively measured on a control bucket running without randomization.

and the average revenue per page, as functions of ρ^*. The inner confidence intervals, represented by the filled areas, grow sharply when ρ^* leaves the range explored during the data collection experiment. The average revenue per page has more variance because a few very competitive queries command high prices.

In order to validate the accuracy of these counterfactual estimates, a second traffic bucket of equal size was configured with mainline reserves reduced by about 18%. The hollow circles in Figure 13 represent the metrics effectively measured on this bucket during the same time period. The effective measurements and the counterfactual estimates match with high accuracy.

Finally, in order to measure the cost of the randomization, we also ran the unmodified ad placement system on a control bucket. The brown filled circles in Figure 13 represent the metrics effectively measured on the control bucket during the same time period. The randomization caused a small but statistically significant increase of the number of mainline ads per page. The click yield and average revenue differences are not significant.

This experiment shows that we can obtain accurate counterfactual estimates with affordable randomization strategies. However, this nice conclusion does not capture the true practical value of the counterfactual estimation approach.

4.7 More on Mainline Reserves

The main benefit of the counterfactual estimation approach is the ability to *use the same data* to answer a *broad range of counterfactual questions*. Here are a few examples of counterfactual questions that can be answered using data collected using the simple mainline reserve randomization scheme described in the previous section:

- *Different variances* – Instead of estimating what would have been measured if we had increased the mainline reserves without changing the randomization variance, that is, letting $\sigma^* = \sigma$, we can use the same data to estimate what would have been measured if we had also changed σ. This provides the means to determine which level of randomization we can afford in future experiments.

- *Pointwise estimates* – We often want to estimate what would have been measured if we had set the mainline reserves to a specific value without randomization. Although computing estimates for small values of σ often works well enough, very small values lead to large confidence intervals.

  Let $Y_\nu(\rho)$ represent the expectation we would have observed if the multipliers m had mean ρ and variance ν. We have then $Y_\nu(\rho) = \mathbb{E}_m[\,\mathbb{E}[y|m]\,] = \mathbb{E}_m[Y_0(m)]$. Assuming that the pointwise value Y_0 is smooth enough for a second order development,

  $$Y_\nu(\rho) \;\approx\; \mathbb{E}_m\big[\,Y_0(\rho) + (m-\rho)Y_0'(\rho) + (m-\rho)^2 Y_0''(\rho)/2\,\big] \;=\; Y_0(\rho) + \nu Y_0''(\rho)/2\,.$$

  Although the reweighting method cannot estimate the point-wise value $Y_0(\rho)$ directly, we can use the reweighting method to estimate both $Y_\nu(\rho)$ and $Y_{2\nu}(\rho)$ with acceptable confidence intervals and write $Y_0(\rho) \approx 2Y_\nu(\rho) - Y_{2\nu}(\rho)$ (Goodwin, 2011).

- *Query-dependent reserves* – Compare for instance the queries "car insurance" and "common cause principle" in a web search engine. Since the advertising potential of a search

varies considerably with the query, it makes sense to investigate various ways to define query-dependent reserves (Charles and Chickering, 2012).

The data collected using the simple mainline reserve randomization can also be used to estimate what would have been measured if we had increased all the mainline reserves by a query-dependent multiplier $\rho^*(x)$. This is simply achieved by reweighting each sample ω_i with

$$w_i = \frac{\mathrm{P}^*(q_i \,|\, x_i, a_i)}{\mathrm{P}(q_i \,|\, x_i, a_i)} = \frac{p(m_i\,;\rho^*(x_i)\,,\sigma)}{p(m_i\,;\mu,\sigma)}\,.$$

Considerably broader ranges of counterfactual questions can be answered when data is collected using randomization schemes that explore more dimensions. For instance, in the case of the ad placement problem, we could apply an independent random multiplier for each score instead of applying a single random multiplier to the mainline reserves only. However, the more dimensions we randomize, the more data needs to be collected to effectively explore all these dimensions. Fortunately, as discussed in section 5, the structure of the causal graph reveals many ways to leverage a priori information and improve the confidence intervals.

4.8 Related Work

Importance sampling is widely used to deal with covariate shifts (Shimodaira, 2000; Sugiyama et al., 2007). Since manipulating the causal graph changes the data distribution, such an intervention can be viewed as a covariate shift amenable to importance sampling. Importance sampling techniques have also been proposed without causal interpretation for many of the problems that we view as causal inference problems. In particular, the work presented in this section is closely related to the Monte-Carlo approach of reinforcement learning (Sutton and Barto, 1998, Chapter 5) and to the offline evaluation of contextual bandit policies (Li et al., 2010, 2011).

Reinforcement learning research traditionally focuses on control problems with relatively small discrete state spaces and long sequences of observations. This focus reduces the need for characterizing exploration with tight confidence intervals. For instance, Sutton and Barto suggest to normalize the importance sampling estimator by $1/\sum_i w(\omega_i)$ instead of $1/n$. This would give erroneous results when the data collection distribution leaves parts of the state space poorly explored. Contextual bandits are traditionally formulated with a finite set of discrete actions. For instance, Li's (2011) unbiased policy evaluation assumes that the data collection policy always selects an arbitrary policy with probability greater than some small constant. This is not possible when the action space is infinite.

Such assumptions on the data collection distribution are often impractical. For instance, certain ad placement policies are not worth exploring because they cannot be implemented efficiently or are known to elicit fraudulent behaviors. There are many practical situations in which one is only interested in limited aspects of the ad placement policy involving continuous parameters such as click prices or reserves. Discretizing such parameters eliminates useful a priori knowledge: for instance, if we slightly increase a reserve, we can reasonable believe that we are going to show slightly less ads.

Instead of making assumptions on the data collection distribution, we construct a biased estimator (10) and bound its bias. We then interpret the inner and outer confidence intervals as resulting from a lack of exploration or an insufficient sample size.

Finally, the causal framework allows us to easily formulate counterfactual questions that pertain to the practical ad placement problem and yet differ considerably in complexity and exploration requirements. We can address specific problems identified by the engineers without incurring the risks associated with a complete redesign of the system. Each of these incremental steps helps demonstrating the soundness of the approach.

5. Structure

This section shows how the structure of the causal graph reveals many ways to leverage a priori knowledge and improve the accuracy of our counterfactual estimates. Displacing the reweighting point (Section 5.1) improves the inner confidence interval and therefore reduce the need for exploration. Using a prediction function (Section 5.2) essentially improve the outer confidence interval and therefore reduce the sample size requirements.

5.1 Better Reweighting Variables

Many search result pages come without eligible ads. We then know with certainty that such pages will have zero mainline ads, receive zero clicks, and generate zero revenue. This is true for the randomly selected value of the reserve, and this would have been true for any other value of the reserve. We can exploit this knowledge by pretending that the reserve was drawn from the counterfactual distribution $\mathrm{P}^*(q\,|\,x_i,a_i)$ instead of the actual distribution $\mathrm{P}(q\,|\,x_i,a_i)$. The ratio $w(\omega_i)$ is therefore forced to the unity. This does not change the estimate but reduces the size of the inner confidence interval. The results of Figure 13 were in fact helped by this little optimization.

There are in fact many circumstances in which the observed outcome would have been the same for other values of the randomized variables. This prior knowledge is in fact encoded in the structure of the causal graph and can be exploited in a more systematic manner. For instance, we know that users make click decisions without knowing which scores were computed by the ad placement engine, and without knowing the prices charged to advertisers. The ad placement causal graph encodes this knowledge by showing the clicks y as direct effects of the user intent u and the ad slate s. This implies that the exact value of the scores q does not matter to the clicks y as long as the ad slate s remains the same.

Because the causal graph has this special structure, we can simplify both the actual and counterfactual Markov factorizations (2) (3) without eliminating the variable y whose expectation is sought. Successively eliminating variables z, c, and q gives:

$$\begin{aligned}
\mathrm{P}(u,v,x,a,b,s,y) &= \mathrm{P}(u,v)\,\mathrm{P}(x\,|\,u)\,\mathrm{P}(a\,|\,x,v)\,\mathrm{P}(b\,|\,x,v)\,\mathrm{P}(s\,|\,x,a,b)\,\mathrm{P}(y\,|\,s,u)\ , \\
\mathrm{P}^*(u,v,x,a,b,s,y) &= \mathrm{P}(u,v)\,\mathrm{P}(x\,|\,u)\,\mathrm{P}(a\,|\,x,v)\,\mathrm{P}(b\,|\,x,v)\,\mathrm{P}^*(s\,|\,x,a,b)\,\mathrm{P}(y\,|\,s,u)\ .
\end{aligned}$$

The conditional distributions $\mathrm{P}(s\,|\,x,a,b)$ and $\mathrm{P}^*(s\,|\,x,a,b)$ did not originally appear in the Markov factorization. They are defined by marginalization as a consequence of the elimination of the variable q representing the scores.

$$\mathrm{P}(s\,|\,x,a,b) = \int_q \mathrm{P}(s\,|\,a,q,b)\,\mathrm{P}(q\,|\,x,a)\ , \quad \mathrm{P}^*(s\,|\,x,a,b) = \int_q \mathrm{P}(s\,|\,a,q,b)\,\mathrm{P}^*(q\,|\,x,a)\,.$$

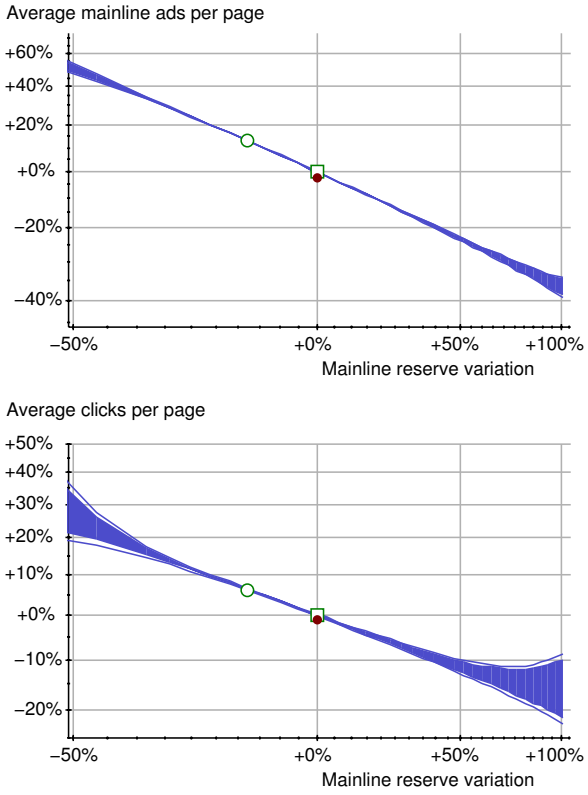


Figure 14: Estimated variations of two performance metrics in response to mainline reserve changes. These estimates were obtained using the ad slates s as reweighting variable. Compare the inner confidence intervals with those shown in Figure 13.

We can estimate the counterfactual click yield Y^* using these simplified factorizations:

$$\begin{aligned} Y^* &= \int y \, \mathrm{P}^*(u,v,x,a,b,s,y) = \int y \, \frac{\mathrm{P}^*(s\,|\,x,a,b)}{\mathrm{P}(s\,|\,x,a,b)} \, \mathrm{P}(u,v,x,a,b,s,y) \\ &\approx \frac{1}{n} \sum_{i=1}^{n} y_i \, \frac{\mathrm{P}^*(s_i\,|\,x_i,a_i,b_i)}{\mathrm{P}(s_i\,|\,x_i,a_i,b_i)} \, . \end{aligned} \tag{16}$$

We have reproduced the experiments described in Section 4.6 with the counterfactual estimate (16) instead of (4). For each example ω_i, we determine which range $[m_i^{\max}, m_i^{\min}]$ of mainline reserve multipliers could have produced the observed ad slate s_i, and then compute the reweighting ratio using the formula:

$$w_i = \frac{\mathrm{P}^*(s_i\,|\,x_i,a_i,b_i)}{\mathrm{P}(s_i\,|\,x_i,a_i,b_i)} = \frac{\Psi(m_i^{\max};\rho^*,\sigma^*) - \Psi(m_i^{\min};\rho^*,\sigma^*)}{\Psi(m_i^{\max};\rho,\sigma) - \Psi(m_i^{\min};\rho,\sigma)} \, ,$$

where $\Psi(m;\rho,\sigma)$ is the cumulative of the log-normal multiplier distribution. Figure 14 shows counterfactual estimates obtained using the same data as Figure 13. The obvious improvement of the

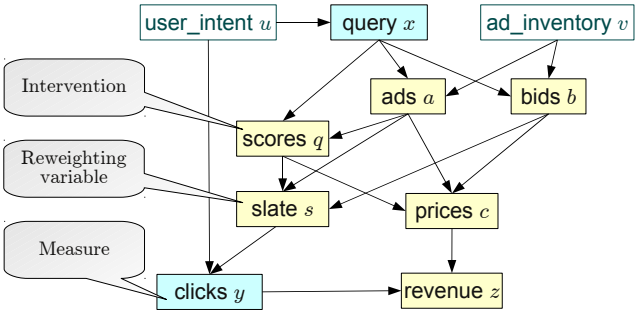


Figure 15: The reweighting variable(s) must intercept all causal paths from the point of intervention to the point of measurement.

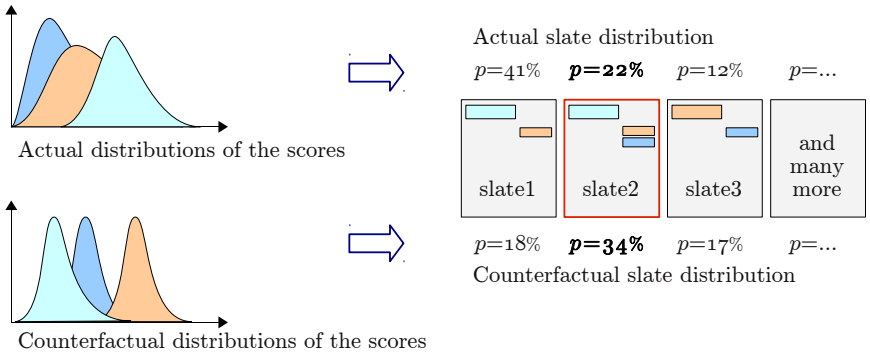


Figure 16: A distribution on the scores q induce a distribution on the possible ad slates s. If the observed slate is `slate2`, the reweighting ratio is $34/22$.

inner confidence intervals significantly extends the range of mainline reserve multipliers for which we can compute accurate counterfactual expectations using this same data.

Comparing (4) and (16) makes the difference very clear: instead of computing the ratio of the probabilities of the observed scores under the counterfactual and actual distributions, we compute the ratio of the probabilities of the observed ad slates under the counterfactual and actual distributions. As illustrated by Figure 15, we now distinguish the reweighting variable (or variables) from the intervention. In general, the corresponding manipulation of the Markov factorization consists of marginalizing out all the variables that appear on the causal paths connecting the point of intervention to the reweighting variables and factoring all the independent terms out of the integral. This simplification works whenever the reweighting variables intercept all the causal paths connecting the point of intervention to the measurement variable. In order to compute the new reweighting ratios, all the factors remaining inside the integral, that is, all the factors appearing on the causal paths connecting the point of intervention to the reweighting variables, have to be known.

Figure 14 does not report the average revenue per page because the revenue z also depends on the scores q through the click prices c. This causal path is not intercepted by the ad slate variable s alone. However, we can introduce a new variable $\tilde{c} = f(c, y)$ that filters out the click prices computed

for ads that did not receive a click. Markedly improved revenue estimates are then obtained by reweighting according to the joint variable $(s, \tilde{c})$.

Figure 16 illustrates the same approach applied to the simultaneous randomization of all the scores q using independent log-normal multipliers. The weight $w(\omega_i)$ is the ratio of the probabilities of the observed ad slate s_i under the counterfactual and actual multiplier distributions. Computing these probabilities amounts to integrating a multivariate Gaussian distribution (Genz, 1992). Details will be provided in a forthcoming publication.

5.2 Variance Reduction with Predictors

Although we do not know exactly how the variable of interest $\ell(\omega)$ depends on the measurable variables and are affected by interventions on the causal graph, we may have strong a priori knowledge about this dependency. For instance, if we augment the slate s with an ad that usually receives a lot of clicks, we can expect an increase of the number of clicks.

Let the *invariant variables* υ be all observed variables that are not direct or indirect effects of variables affected by the intervention under consideration. This definition implies that the distribution of the invariant variables is not affected by the intervention. Therefore the values υ_i of the invariant variables sampled during the actual experiment are also representative of the distribution of the invariant variables under the counterfactual conditions.

We can leverage a priori knowledge to construct a predictor $\zeta(\omega)$ of the quantity $\ell(\omega)$ whose counterfactual expectation Y^* is sought. We assume that the predictor $\zeta(\omega)$ depends only on the invariant variables or on variables that depend on the invariant variables through known functional dependencies. Given sampled values υ_i of the invariant variables, we can replay both the original and manipulated structural equation model as explained in Section 4.1 and obtain samples ζ_i and ζ_i^* that respectively follow the actual and counterfactual distributions

Then, regardless of the quality of the predictor,

$$\begin{aligned} Y^* \;=\; \int_\omega \ell(\omega)\,\mathrm{P}^*(\omega) \;&=\; \int_\omega \zeta(\omega)\,\mathrm{P}^*(\omega) \;+\; \int_\omega (\ell(\omega)-\zeta(\omega))\,\mathrm{P}^*(\omega) \\ &\approx\; \frac{1}{n}\sum_{i=1}^n \zeta_i^* \;+\; \frac{1}{n}\sum_{i=1}^n (\ell(\omega_i)-\zeta_i)\,w(\omega_i)\,. \end{aligned} \tag{17}$$

The first term in this sum represents the counterfactual expectation of the predictor and can be accurately estimated by averaging the simulated counterfactual samples ζ_i^* without resorting to potentially large importance weights. The second term in this sum represents the counterfactual expectation of the residuals $\ell(\omega)-\zeta(\omega)$ and must be estimated using importance sampling. Since the magnitude of the residuals is hopefully smaller than that of $\ell(\omega)$, the variance of $(\ell(\omega)-\zeta(\omega))\,w(\omega)$ is reduced and the importance sampling estimator of the second term has improved confidence intervals. The more accurate the predictor $\zeta(\omega)$, the more effective this variance reduction strategy.

This variance reduction technique is in fact identical to the doubly robust contextual bandit evaluation technique of Dudík et al. (2012). Doubly robust variance reduction has also been extensively used for causal inference applied to biostatistics (see Robins et al., 2000; Bang and Robins, 2005). We subjectively find that viewing the predictor as a component of the causal graph (Figure 17) clarifies how a well designed predictor can leverage prior knowledge. For instance, in order to estimate the counterfactual performance of the ad placement system, we can easily use a predictor that runs the ad auction and simulate the user clicks using a click probability model trained offline.

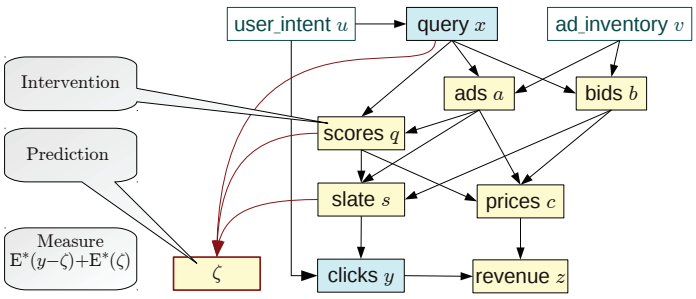


Figure 17: Leveraging a predictor. Yellow nodes represent known functional relations in the structural equation model. We can estimate the counterfactual expectation Y^* of the number of clicks per page as the sum of the counterfactual expectations of a predictor ζ, which is easy to estimate by replaying empirical data, and $y - \zeta$, which has to be estimated by importance sampling but has reduced variance.

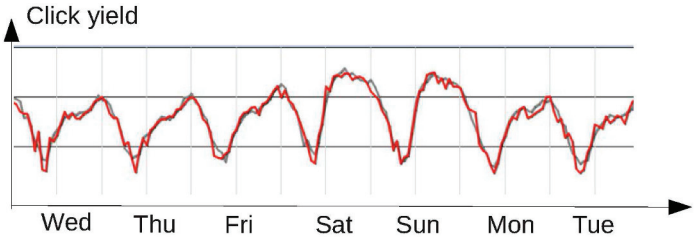


Figure 18: The two plots show the hourly click yield for two variants of the ad placement engine. The daily variations dwarf the differences between the two treatments.

5.3 Invariant Predictors

In order to evaluate which of two interventions is most likely to improve the system, the designer of a learning system often seeks to estimate a *counterfactual difference*, that is, the difference $Y^+ - Y^*$ of the expectations of a same quantity $\ell(\omega)$ under two different counterfactual distributions $\mathrm{P}^+(\omega)$ and $\mathrm{P}^*(\omega)$. These expectations are often affected by variables whose value is left unchanged by the interventions under consideration. For instance, seasonal effects can have very large effects on the number of ad clicks (Figure 18) but affect Y^+ and Y^* in similar ways.

Substantially better confidence intervals on the difference $Y^+ - Y^*$ can be obtained using an *invariant predictor*, that is, a predictor function that depends only on invariant variables υ such as the time of the day. Since the invariant predictor $\zeta(\upsilon)$ is not affected by the interventions under consideration,

$$\int_\omega \zeta(\upsilon)\,\mathrm{P}^*(\omega) = \int_\omega \zeta(\upsilon)\,\mathrm{P}^+(\omega)\,. \tag{18}$$

Therefore

$$\begin{aligned} Y^+ - Y^* &= \int_\omega \zeta(\upsilon)\,\mathrm{P}^+(\omega) + \int_\omega (\ell(\omega) - \zeta(\upsilon))\,\mathrm{P}^+(\omega) \\ &\qquad - \int_\omega \zeta(\upsilon)\,\mathrm{P}^*(\omega) - \int_\omega (\ell(\omega) - \zeta(\upsilon))\,\mathrm{P}^*(\omega) \\ &\approx \frac{1}{n}\sum_{i=1}^{n} \big(\ell(\omega_i) - \zeta(\upsilon_i)\big)\,\frac{P^+(\omega_i) - P^*(\omega_i)}{P(\omega_i)}\,. \end{aligned}$$

This direct estimate of the counterfactual difference $Y^+ - Y^*$ benefits from the same variance reduction effect as (17) without need to estimate the expectations (18). Appendix C provide details on the computation of confidence intervals for estimators of the counterfactual differences. Appendix D shows how the same approach can be used to compute *counterfactual derivatives* that describe the response of the system to very small interventions.

6. Learning

The previous sections deal with the identification and the measurement of interpretable signals that can justify the actions of human decision makers. These same signals can also justify the actions of machine learning algorithms. This section explains why optimizing a counterfactual estimate is a sound learning procedure.

6.1 A Learning Principle

We consider in this section interventions that depend on a parameter θ. For instance, we might want to know what the performance of the ad placement engine would have been if we had used different values for the parameter θ of the click scoring model. Let $\mathrm{P}^\theta(\omega)$ denote the counterfactual Markov factorization associated with this intervention. Let Y^θ be the counterfactual expectation of $\ell(\omega)$ under distribution P^θ. Figure 19 illustrates our simple learning setup. Training data is collected from a single experiment associated with an initial parameter value θ^0 chosen using prior knowledge acquired in an unspecified manner. A preferred parameter value θ^* is then determined using the training data and loaded into the system. The goal is of course to observe a good performance on data collected during a test period that takes place after the switching point.

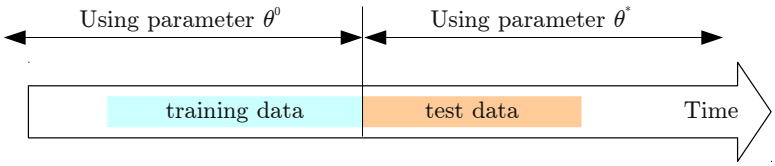


Figure 19: Single design – A preferred parameter value θ^* is determined using randomized data collected in the past. Test data is collected after loading θ^* into the system.

The isolation assumption introduced in Section 3.2 states that the exogenous variables are drawn from an unknown but fixed joint probability distribution. This distribution induces a joint distribution $P(\omega)$ on all the variables ω appearing in the structural equation model associated with the parameter θ. Therefore, if the *isolation assumption remains valid during the test period*, the test data follows the same distribution $P^{\theta^*}(\omega)$ that would have been observed during the training data collection period if the system had been using parameter θ^* all along.

We can therefore formulate this problem as the optimization of the expectation Y^θ of the reward $\ell(\omega)$ with respect to the distribution $P^\theta(\omega)$

$$\max_\theta Y^\theta = \int_\omega \ell(\omega)\, P^\theta(\omega)$$

on the basis of a finite set of training examples $\omega_1, \ldots, \omega_n$ sampled from $P(\omega)$.

However, it would be unwise to maximize the estimates obtained using approximation (5) because they could reach a maximum for a value of θ that is poorly explored by the actual distribution. As explained in Section 4.5, the gap between the upper and lower bound of inequality (14) reveals the uncertainty associated with insufficient exploration. Maximizing an empirical estimate $\widehat{Y}^\theta$ of the lower bound $\bar{Y}^\theta$ ensures that the optimization algorithm finds a trustworthy answer

$$\theta^* = \arg\max_\theta \widehat{Y}^\theta \,. \tag{19}$$

We shall now discuss the statistical basis of this learning principle.[9]

6.2 Uniform Confidence Intervals

As discussed in Section 4.4, inequality (14),

$$\bar{Y}^\theta \le Y^\theta \le \bar{Y}^\theta + M(1-\bar{W}^\theta) \,,$$

where

$$\bar{Y}^\theta = \int_\omega \ell(\omega)\,\bar{w}(\omega)\,P(\omega) \approx \widehat{Y}^\theta = \frac{1}{n}\sum_{i=1}^n \ell(\omega_i)\,\bar{w}(\omega_i) \,,$$
$$\bar{W}^\theta = \int_\omega \bar{w}(\omega)\,P(\omega) \approx \widehat{W}^\theta = \frac{1}{n}\sum_{i=1}^n \bar{w}(\omega_i) \,,$$

9. The idea of maximizing the lower bound may surprise readers familiar with the UCB algorithm for multi-armed bandits (Auer et al., 2002). UCB performs exploration by maximizing the upper confidence interval bound and updating the confidence intervals online. Exploration in our setup results from the active system randomization during the offline data collection. See also Section 6.4.

leads to confidence intervals (15) of the form

$$\forall \delta > 0,\ \forall \theta \quad \mathbb{P}\Big\{ \widehat{Y}^\theta - \varepsilon_R \ \le\ Y^\theta \ \le\ \widehat{Y}^\theta + M(1 - \widehat{W}^\theta + \xi_R) + \varepsilon_R \Big\} \ge 1 - \delta . \tag{20}$$

Both ε_R and ξ_R converge to zero in inverse proportion to the square root of the sample size n. They also increase at most linearly in $\log\delta$ and depend on both the capping bound R and the parameter θ through the empirical variances (see appendix B.)

Such confidence intervals are insufficient to provide guarantees for a parameter value θ^* that depends on the sample. In fact, the optimization (19) procedure is likely to select values of θ for which the inequality is violated. We therefore seek uniform confidence intervals (Vapnik and Chervonenkis, 1968), simultaneously valid for all values of θ.

- When the parameter θ is chosen from a finite set $\mathcal{F}$, applying the union bound to the ordinary intervals (20) immediately gives the uniform confidence interval :

$$\mathbb{P}\Big\{ \forall \theta \in \mathcal{F},\ \widehat{Y}^\theta - \varepsilon_R \le Y^\theta \le \widehat{Y}^\theta + M(1 - \widehat{W}^\theta + \xi_R) + \varepsilon_R \Big\} \ge 1 - |\mathcal{F}|\,\delta .$$

- Following the pioneering work of Vapnik and Chervonenkis, a broad choice of mathematical tools have been developed to construct uniform confidence intervals when the set $\mathcal{F}$ is infinite. For instance, appendix E leverages uniform empirical Bernstein bounds (Maurer and Pontil, 2009) and obtains the uniform confidence interval

$$\mathbb{P}\Big\{ \forall \theta \in \mathcal{F},\ \widehat{Y}^\theta - \varepsilon_R \le Y^\theta \le \widehat{Y}^\theta + M(1 - \widehat{W}^\theta + \xi_R) + \varepsilon_R \Big\} \ge 1 - \mathcal{M}(n)\,\delta , \tag{21}$$

  where the growth function $\mathcal{M}(n)$ measures the capacity of the family of functions

$$\{\ f_\theta : \omega \mapsto \ell(\omega)\bar{w}(\omega)\ ,\ \ g_\theta : \omega \mapsto \bar{w}(\omega)\ ,\ \ \forall \theta \in \mathcal{F}\ \} .$$

  Many practical choices of $P^*(\omega)$ lead to functions $\mathcal{M}(n)$ that grow polynomially with the sample size. Because both ε_R and ξ_R are $O(n^{-1/2}\log\delta)$, they converge to zero with the sample size when one maintains the confidence level $1 - \mathcal{M}(n)\,\delta$ equal to a predefined constant.

The interpretation of the inner and outer confidence intervals (Section 4.5) also applies to the uniform confidence interval (21). When the sample size is sufficiently large and the capping bound R chosen appropriately, the inner confidence interval reflects the upper and lower bound of inequality (14).

The uniform confidence interval therefore ensures that Y^{θ^*} is close to the maximum of the lower bound of inequality (14) which essentially represents the best performance that can be guaranteed using training data sampled from $P(\omega)$. Meanwhile, the upper bound of this same inequality reveals which values of θ could potentially offer better performance but have been insufficiently probed by the sampling distribution (Figure 20.)

6.3 Tuning Ad Placement Auctions

We now present an application of this learning principle to the optimization of auction tuning parameters in the ad placement engine. Despite increasingly challenging engineering difficulties,

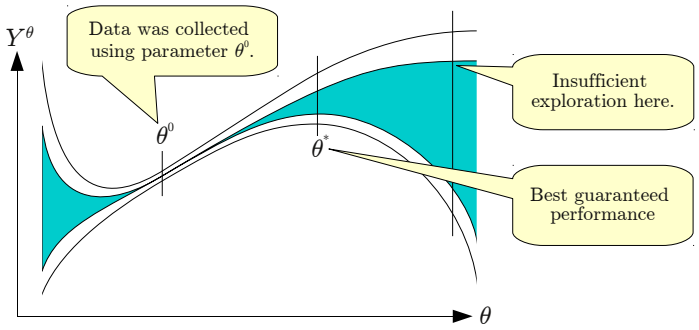


Figure 20: The uniform inner confidence interval reveals where the best guaranteed Y^θ is reached and w

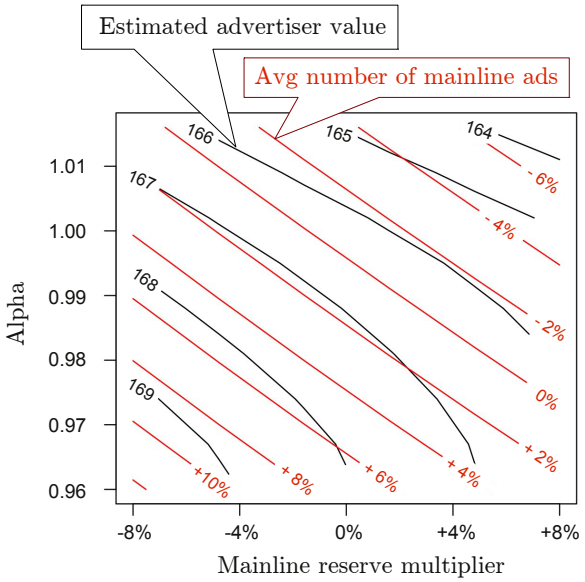


Figure 21: Level curves associated with the average number of mainline ads per page (red curves labelled from -6% to $+10\%$) and the average estimated advertisement value generated per page (black curves, labelled with arbitrary units ranging from 164 to 169) that would have been observed for a certain query cluster if we had changed the mainline reserves by the multiplicative factor shown on the horizontal axis, and if we had applied a squashing exponent α shown on the vertical axis to the estimated click probabilities $q_{i,p}(x)$.

comparable optimization procedures can obviously be applied to larger numbers of tunable parameters.

Lahaie and McAfee (2011) propose to account for the uncertainty of the click probability estimation by introducing a squashing exponent α to control the impact of the estimated probabilities on the rank scores. Using the notations introduced in Section 2.1, and assuming that the estimated probability of a click on ad i placed at position p after query x has the form $q_{ip}(x) = \gamma_p \beta_i(x)$ (see

appendix A), they redefine the rank-score $r_{ip}(x)$ as:

$$r_{ip}(x) = \gamma_p \, b_i \, \beta_i(x)^{\alpha} \, .$$

Using a squashing exponent $\alpha < 1$ reduces the contribution of the estimated probabilities and increases the reliance on the bids b_i placed by the advertisers.

Because the squashing exponent changes the rank-score scale, it is necessary to simultaneously adjust the reserves in order to display comparable number of ads. In order to estimate the counterfactual performance of the system under interventions affecting both the squashing exponent and the mainline reserves, we have collected data using a random squashing exponent following a normal distribution, and a mainline reserve multiplier following a log-normal distribution as described in Section 4.6. Samples describing 12 million search result pages were collected during four consecutive weeks.

Following Charles and Chickering (2012), we consider separate squashing coefficients α_k and mainline reserve multipliers ρ_k per query cluster $k \in \{1..K\}$, and, in order to avoid negative user or advertiser reactions, we seek the auction tuning parameters α_k and ρ_k that maximize an estimate of the advertisement value[10] subject to a global constraint on the average number of ads displayed in the mainline. Because maximizing the advertisement value instead of the publisher revenue amounts to maximizing the size of the advertisement pie instead of the publisher slice of the pie, this criterion is less likely to simply raise the prices without improving the ads. Meanwhile the constraint ensures that users are not exposed to excessive numbers of mainline ads.

We then use the collected data to estimate bounds on the counterfactual expectations of the advertiser value and the counterfactual expectation of the number of mainline ads per page. Figure 21 shows the corresponding level curves for a particular query cluster. We can then run a simple optimization algorithm and determine the optimal auction tuning parameters for each cluster subject to the global mainline footprint constraint. Appendix D describes how to estimate off-policy counterfactual derivatives that greatly help the numerical optimization.

The obvious alternative (see Charles and Chickering, 2012) consists of replaying the auctions with different parameters and simulating the user using a click probability model. However, it may be unwise to rely on a click probability model to estimate the best value of a squashing coefficient that is expected to compensate for the uncertainty of the click prediction model itself. The counterfactual approach described here avoids the problem because it does not rely on a click prediction model to simulate users. Instead it estimates the counterfactual performance of the system using the actual behavior of the users collected under moderate randomization.

6.4 Sequential Design

Confidence intervals computed after a first randomized data collection experiment might not offer sufficient accuracy to choose a final value of the parameter θ. It is generally unwise to simply collect additional samples using the same experimental setup because the current data already reveals information (Figure 20) that can be used to design a better data collection experiment. Therefore, it seems natural to extend the learning principle discussed in Section 6.1 to a sequence of data collection experiments. The parameter θ_t characterizing the t-th experiment is then determined using samples collected during the previous experiments (Figure 22).

10. The value of an ad click from the point of view of the advertiser. The advertiser payment then splits the advertisement value between the publisher and the advertiser.

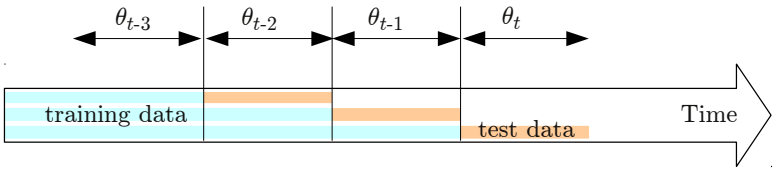


Figure 22: Sequential design – The parameter θ_t of each data collection experiment is determined using data collected during the previous experiments.

Although it is relatively easy to construct convergent sequential design algorithms, reaching the *optimal* learning performance is notoriously difficult (Wald, 1945) because the selection of parameter θ_t involves a trade-off between exploitation, that is, the maximization of the immediate reward Y^{θ_t}, and exploration, that is, the collection of samples potentially leading to better Y^{θ} in the more distant future.

The optimal exploration exploitation trade-off for multi-armed bandits is well understood (Gittins, 1989; Auer et al., 2002; Audibert et al., 2007) because an essential property of multi-armed bandits makes the analysis much simpler: the outcome observed after performing a particular action brings no information about the value of other actions. Such an assumption is both unrealistic and pessimistic. For instance, the outcome observed after displaying a certain ad in response to a certain query brings very useful information about the value of displaying similar ads on similar queries.

Refined contextual bandit approaches (Slivkins, 2011) account for similarities in the context and action spaces but do not take advantage of all the additional opportunities expressed by structural equation models. For instance, in the contextual bandit formulation of the ad placement problem outlined in Section 3.5, actions are pairs (s,c) describing the ad slate s and the corresponding click prices c, policies select actions by combining individual ad scores in very specific ways, and actions determine the rewards through very specific mechanisms.

Meanwhile, despite their suboptimal asymptotic properties, heuristic exploration strategies perform surprisingly well during the time span in which the problem can be considered stationary. Even in the simple case of multi-armed bandits, excellent empirical results have been obtained using Thompson sampling (Chapelle and Li, 2011) or fixed strategies (Vermorel and Mohri, 2005; Kuleshov and Precup, 2010). Leveraging the problem structure seems more important in practice than perfecting an otherwise sound exploration strategy.

Therefore, in the absence of sufficient theoretical guidance, it is both expedient and practical to maximizing $\widehat{Y}^{\theta}$ at each round, as described in Section 6.1, subject to additional ad-hoc constraints ensuring a minimum level of exploration.

7. Equilibrium Analysis

All the methods discussed in this contribution rely on the isolation assumption presented in Section 3.2. This assumption lets us interpret the samples as repeated independent trials that follow the pattern defined by the structural equation model and are amenable to statistical analysis.

The isolation assumption is in fact a component of the counterfactual conditions under investigation. For instance, in Section 4.6, we model single auctions (Figure 3) in order to empirically

determine how the ad placement system would have performed if we had changed the mainline reserves *without incurring a reaction from the users or the advertisers*.

Since the future publisher revenues depend on the continued satisfaction of users and advertisers, lifting this restriction is highly desirable.

- We can in principle work with larger structural equation models. For instance, Figure 4 suggests to thread single auction models with additional causal links representing the impact of the displayed ads on the future user goodwill. However, there are practical limits on the number of trials we can consider at once. For instance, it is relatively easy to simultaneously model all the auctions associated with the web pages served to the same user during a thirty minute web session. On the other hand, it is practically impossible to consider several weeks worth of auctions in order to model their accumulated effect on the continued satisfaction of users and advertisers.

- We can sometimes use problem-specific knowledge to construct alternate performance metrics that anticipate the future effects of the feedback loops. For instance, in Section 6.3, we optimize the advertisement value instead of the publisher revenue. Since this alternative criterion takes the advertiser interests into account, it can be viewed as a heuristic proxy for the future revenues of the publisher.

This section proposes an alternative way to account for such feedback loops using the *quasistatic equilibrium* method familiar to physicists: we assume that the publisher changes the parameter θ so slowly that the system remains at equilibrium at all times. Using data collected while the system was at equilibrium, we describe empirical methods to determine how an infinitesimal intervention $d\theta$ on the model parameters would have displaced the equilibrium:

> "*How would the system have performed during the data collection period if a small change* $d\theta$ *had been applied to the model parameter* θ *and the equilibrium had been reached before the data collection period.*"

A learning algorithm can then update θ to improve selected performance metrics.

7.1 Rational Advertisers

The ad placement system is an example of game where each actor furthers his or her interests by controlling some aspects of the system: the publisher controls the placement engine parameters, the advertisers control the bids, and the users control the clicks.

As an example of the general quasi-static approach, this section focuses on the reaction of *rational advertisers* to small changes of the scoring functions driving the ad placement system. Rational advertisers always select bids that maximize their economic interests. Although there are more realistic ways to model advertisers, this exercise is interesting because the auction theory approaches also rely on the rational advertiser assumption (see Section 2.1). This analysis seamlessly integrates the auction theory and machine learning perspectives.

As illustrated in Figure 23, we treat the bid vector $b_\star = (b_1 \dots b_A) \in [0, b_{\max}]^A$ as the parameter of the conditional distribution $P^{b_\star}(b|x,v)$ of the bids associated with the eligible ads.[11] The vari-

11. Quantities measured when a feedback causal system reaches equilibrium often display conditional independence patterns that cannot be represented with directed acyclic graphs (Lauritzen and Richardson, 2002; Dash, 2003). Treating the feedback loop as parameters instead of variables works around this difficulty in a manner that appears sufficient to perform the quasi-static analysis.

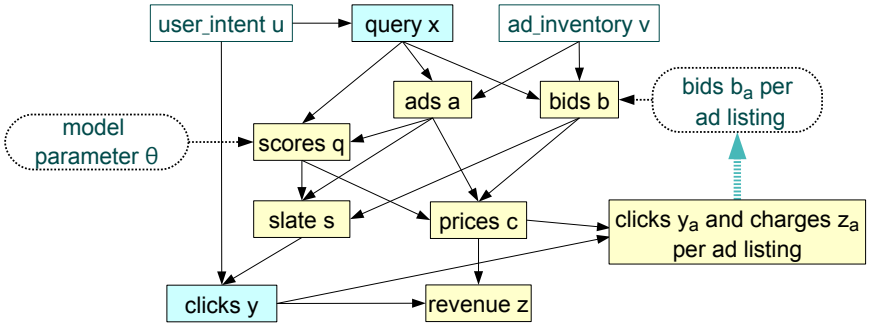


Figure 23: Advertisers select the bid amounts b_a on the basis of the past number of clicks y_a and the past prices z_a observed for the corresponding ads.

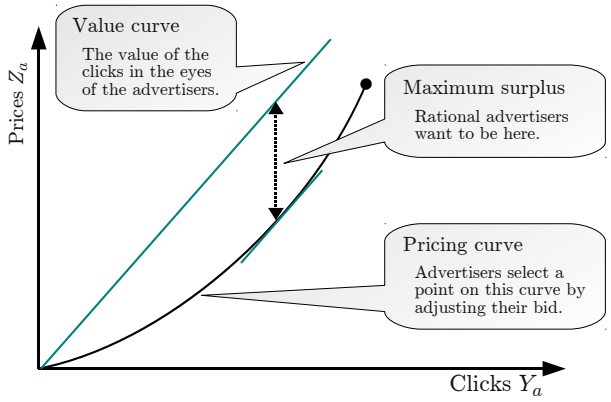


Figure 24: Advertisers control the expected number of clicks Y_a and expected prices Z_a by adjusting their bids b_a. Rational advertisers select bids that maximize the difference between the value they see in the clicks and the price they pay.

ables y_a in the structural equation model represents the number of clicks received by ads associated with bid b_a. The variables z_a represents the amount charged for these clicks to the corresponding advertiser. The advertisers select their bids b_a according to their anticipated impact on the number of resulting clicks y_a and on their cost z_a.

Following the pattern of the perfect information assumption (see Section 2.1), we assume that the advertisers eventually acquire full knowledge of the expectations

$$Y_a(\theta, b_\star) = \int_\omega y_a \, \mathrm{P}^{\theta, b_\star}(\omega) \quad \text{and} \quad Z_a(\theta, b_\star) = \int_\omega z_a \, \mathrm{P}^{\theta, b_\star}(\omega) \, .$$

Let V_a denote the value of a click for the corresponding advertiser. Rational advertiser seek to maximize the difference between the value they see in the clicks and the price they pay to the publisher, as illustrated in Figure 24. This is expressed by the utility functions

$$U_a^\theta(b_\star) \;=\; V_a Y_a(\theta, b_\star) - Z_a(\theta, b_\star) \, .$$

Following Athey and Nekipelov (2010), we argue that the injection of smooth random noise into the auction mechanism changes the discrete problem into a continuous problem amenable to standard differential methods. Mild regularity assumption on the densities probability $\mathrm{P}^{b_\star}(b|x,v)$ and $\mathrm{P}^{\theta}(q|x,a)$ are in fact sufficient to ensure that the expectations $Y_a(\theta,b_\star)$ and $Z_a(\theta,b_\star)$ are continuously differentiable functions of the distribution parameters $b_\star$ and θ. Further assuming that utility functions $U_a^\theta(b_\star)$ are diagonally quasiconcave, Athey and Nekipelov establish the existence of a unique Nash equilibrium

$$\forall a \quad b_a \in \operatorname*{Arg\,Max}_b \; U_a^\theta(b_1,\ldots,b_{a-1},b,b_{a+1},\ldots,b_A)$$

characterized by its first order Karush-Kuhn-Tucker conditions

$$\forall a \qquad V_a \frac{\partial Y_a}{\partial b_a} - \frac{\partial Z_a}{\partial b_a} \quad \begin{cases} \le 0 & \text{if } b_a = 0, \\ \ge 0 & \text{if } b_a = b_{\max}, \\ = 0 & \text{if } 0 < b_a < b_{\max}. \end{cases} \tag{22}$$

We use the first order equilibrium conditions (22) for two related purposes. Section 7.2 explains how to complete the advertiser model by estimating the values V_a. Section 7.3 estimates how the equilibrium bids and the system performance metrics respond to a small change $\mathrm{d}\theta$ of the model parameters.

Interestingly, this approach remains sensible when key assumptions of the equilibrium model are violated. The perfect information assumption is unlikely to hold in practice. The quasi-concavity of the utility functions is merely plausible. However, after observing the operation of the stationary ad placement system for a sufficiently long time, it is reasonable to assume that the most active advertisers have tried small bid variations and have chosen locally optimal ones. Less active advertisers may leave their bids unchanged for longer time periods, but can also update them brutally if they experience a significant change in return on investment. Therefore it makes sense to use data collected when the system is stationary to estimate advertiser values V_a that are consistent with the first order equilibrium conditions. We then hope to maintain the conditions that each advertisers had found sufficiently attractive, by first estimating how a small change $\mathrm{d}\theta$ displaces this posited local equilibrium, then by using performance metrics that take this displacement into account.

7.2 Estimating Advertiser Values

We first need to estimate the partial derivatives appearing in the equilibrium condition (22). These derivatives measure how the expectations Y_a and Z_a would have been changed if each advertiser had placed a slightly different bid b_a. Such quantities can be estimated by randomizing the bids and computing on-policy counterfactual derivatives as explained in appendix D. Confidence intervals can be derived with the usual tools.

Unfortunately, the publisher is not allowed to directly randomize the bids because the advertisers expect to pay prices computed using the bid they have specified and not the potentially higher bids resulting from the randomization. However, the publisher has full control on the estimated click probabilities $q_{i,p}(x)$. Since the rank-scores $r_{i,p}(x)$ are the products of the bids and the estimated click probabilities (see Section 2.1), a random multiplier applied to the bids can also be interpreted as a random multiplier applied to the estimated click probabilities. Under these two interpretations, the same ads are shown to the users, but different click prices are charged to the advertisers. Therefore,

the publisher can simultaneously charge prices computed as if the multiplier had been applied to the estimated click probabilities, and collect data as if the multiplier had been applied to the bid. This data can then be used to estimate the derivatives.

Solving the first order equilibrium equations then yields estimated advertiser values V_a that are consistent with the observed data.[12]

$$V_a \approx \frac{\partial Y_a}{\partial b_a} \Big/ \frac{\partial Z_a}{\partial b_a}$$

There are however a couple caveats:

- The advertiser bid b_a may be too small to cause ads to be displayed. In the absence of data, we have no means to estimate a click value for these advertisers.
- Many ads are not displayed often enough to obtain accurate estimates of the partial derivatives $\frac{\partial Y_a}{\partial b_a}$ and $\frac{\partial Z_a}{\partial b_a}$. This can be partially remediated by smartly aggregating the data of advertisers deemed similar.
- Some advertisers attempt to capture all the available ad opportunities by placing extremely high bids and hoping to pay reasonable prices thanks to the generalized second price rule. Both partial derivatives $\frac{\partial Y_a}{\partial b_a}$ and $\frac{\partial Z_a}{\partial b_a}$ are equal to zero in such cases. Therefore we cannot recover V_a by solving the equilibrium Equation (22). It is however possible to collect useful data by selecting for these advertisers a maximum bid $b_{\max}$ that prevents them from monopolizing the eligible ad opportunities. Since the equilibrium condition is an inequality when $b_a = b_{\max}$, we can only determine a lower bound of the values V_a for these advertisers.

These caveats in fact underline the limitations of the advertiser modelling assumptions. When their ads are not displayed often enough, advertisers have no more chance to acquire a full knowledge of the expectations Y_a and Z_a than the publisher has a chance to determine their value. Similarly, advertisers that place extremely high bids are probably underestimating the risk to occasionally experience a very high click price. A more realistic model of the advertiser information acquisition is required to adequately handle these cases.

7.3 Estimating the Equilibrium Response

Let $\mathcal{A}$ be the set of the *active advertisers*, that is, the advertisers whose value can be estimated (or lower bounded) with sufficient accuracy. Assuming that the other advertisers leave their bids unchanged, we can estimate how the active advertisers adjust their bids in response to an infinitesimal change $\mathrm{d}\theta$ of the scoring model parameters. This is achieved by differentiating the equilibrium equations (22):

$$\forall a' \in \mathcal{A}, \quad 0 \;=\; \left(V_{a'} \frac{\partial^2 Y_{a'}}{\partial b_{a'} \partial \theta} - \frac{\partial^2 Z_{a'}}{\partial b_{a'} \partial \theta} \right) \mathrm{d}\theta + \sum_{a \in \mathcal{A}} \left(V_{a'} \frac{\partial^2 Y_{a'}}{\partial b_{a'} \partial b_a} - \frac{\partial^2 Z_{a'}}{\partial b_{a'} \partial b_a} \right) \mathrm{d}b_a \,. \qquad (23)$$

The partial second derivatives must be estimated as described in appendix D. Solving this linear system of equations then yields an expression of the form

$$\mathrm{d}b_a \;=\; \Xi_a \, \mathrm{d}\theta \,.$$

12. This approach is of course related to the value estimation method proposed by Athey and Nekipelov (2010) but strictly relies on the explicit randomization of the scores. In contrast, practical considerations force Athey and Nekipelov to rely on the apparent noise and hope that the noise model accounts for all potential confounding factors.

This expression can then be used to estimate how any counterfactual expectation Y of interest changes when the publisher applies an infinitesimal change $\mathrm{d}\theta$ to the scoring parameter θ and the active advertisers $\mathcal{A}$ rationally adjust their bids b_a in response:

$$\mathrm{d}Y = \left(\frac{\partial Y}{\partial \theta} + \sum_a \Xi_a \frac{\partial Y}{\partial b_a} \right) \mathrm{d}\theta \; . \tag{24}$$

Although this expression provides useful information, one should remain aware of its limitations. Because we only can estimate the reaction of active advertisers, expression (24) does not includes the potentially positive reactions of advertisers who did not bid but could have. Because we only can estimate a lower bound of their values, this expression does not model the potential reactions of advertisers placing unrealistically high bids. Furthermore, one needs to be very cautious when the system (23) approaches singularities. Singularities indicate that the rational advertiser assumption is no longer sufficient to determine the reactions of certain advertisers. This happens for instance when advertisers cannot find bids that deliver a satisfactory return. The eventual behavior of such advertisers then depends on factors not taken in consideration by our model.

To alleviate these issues, we could alter the auction mechanism in a manner that forces advertisers to reveal more information, and we could enforce policies ensuring that the system (23) remains safely nonsingular. We could also design experiments revealing the impact of the fixed costs incurred by advertisers participating into new auctions. Although additional work is needed to design such refinements, the quasistatic equilibrium approach provides a generic framework to take such aspects into account.

7.4 Discussion

The rational advertiser assumption is the cornerstone of seminal works describing simplified variants of the ad placement problem using auction theory (Varian, 2007; Edelman et al., 2007). More sophisticated works account for more aspects of the ad placement problem, such as the impact of click prediction learning algorithms (Lahaie and McAfee, 2011), the repeated nature of the ad auctions (Bergemann and Said, 2010), or for the fact that advertisers place bids valid for multiple auctions (Athey and Nekipelov, 2010). Despite these advances, it seems technically very challenging to use these methods and account for all the effects that can be observed in practical ad placement systems.

We believe that our counterfactual reasoning framework is best viewed as a modular toolkit that lets us apply insights from auction theory and machine learning to problems that are far more complex than those studied in any single paper. For instance, the quasi-static equilibrium analysis technique illustrated in this section extends naturally to the analysis of multiple simultaneous causal feedback loops involving additional players:

- The first step consists in designing ad-hoc experiments to identify the parameters that determine the equilibrium equation of each player. In the case of the advertisers, we have shown how to use randomized scores to reveal the advertiser values. In the case of the user feedback, we must carefully design experiments that reveal how users respond to changes in the quality of the displayed ads.

- Differentiating all the equilibrium equations yields a linear system of equations linking the variations of the parameter under our control, such as $d\theta$, and all the parameters under the

control of the other players, such as the advertiser bids, or the user willingness to visit the site and click on ads. Solving this system and writing the total derivative of the performance measure gives the answer to our question.

Although this programme has not yet been fully realized, the existence of a principled framework to handle such complex interactions is remarkable. Furthermore, thanks to the flexibility of the causal inference frameworks, these techniques can be infinitely adapted to various modeling assumptions and various system complexities.

8. Conclusion

Using the ad placement example, this work demonstrates the central role of causal inference (Pearl, 2000; Spirtes et al., 1993) for the design of learning systems interacting with their environment. Thanks to importance sampling techniques, data collected during randomized experiments gives precious cues to assist the designer of such learning systems and useful signals to drive learning algorithms.

Two recurrent themes structure this work. First, we maintain a sharp distinction between the learning algorithms and the extraction of the signals that drive them. Since real world learning systems often involve a mixture of human decision and automated processes, it makes sense to separate the discussion of the learning signals from the discussion of the learning algorithms that leverage them. Second, we claim that the mathematical and philosophical tools developed for the analysis of physical systems appear very effective for the analysis of causal information systems and of their equilibria. These two themes are in fact a vindication of cybernetics (Wiener, 1948).

Acknowledgments

We would like to acknowledge extensive discussions with Susan Athey, Miroslav Dudík, Patrick Jordan, John Langford, Lihong Li, Sebastien Lahaie, Shie Mannor, Chris Meek, Alex Slivkins, and Paul Viola. We also thank the Microsoft adCenter RnR team for giving us the invaluable opportunity to deploy these ideas at scale and prove their worth. Finally we gratefully acknowledge the precious comments of our JMLR editor and reviewers.

Appendix A. Greedy Ad Placement Algorithms

Section 2.1 describes how to select and place ads on a web page by maximizing the total rank-score (1). Following (Varian, 2007; Edelman et al., 2007) , we assume that the click probability estimates are expressed as the product of a positive position term γ_p and a positive ad term $\beta_i(x)$. The rank-scores can therefore be written as $r_{i,p}(x) = \gamma_p b_i \beta_i(x)$. We also assume that the policy constraints simply state that a web page should not display more than one ad belonging to any given advertiser. The discrete maximization problem is then amenable to computationally efficient greedy algorithms.

Let us fix a layout L and focus on the inner maximization problem. Without loss of generality, we can renumber the positions such that

$$L = \{1, 2, \ldots N\} \quad \text{and} \quad \gamma_1 \geq \gamma_2 \geq \cdots \geq 0\,.$$

and write the inner maximization problem as

$$\max_{i_1,\ldots,i_N} \mathcal{R}_L(i_1,\ldots,i_N) \;=\; \sum_{p\in L} r_{i_p,p}(x)$$

subject to the policy constraints and reserve constraints $r_{i,p}(x) \geq R_p(x)$.

Let S_i denote the advertiser owning ad i. The set of ads is then partitioned into subsets $I_s = \{i : S_i = s\}$ gathering the ads belonging to the same advertiser s. The ads that maximize the product $b_i\beta_i(x)$ within set I_s are called the best ads for advertiser s. If the solution of the discrete maximization problem contains one ad belonging to advertiser s, then it is easy to see that this ad must be one of the best ads for advertiser s: were it not the case, replacing the offending ad by one of the best ads would yield a higher $\mathcal{R}_L$ without violating any of the constraints. It is also easy to see that one could select any of the best ads for advertiser s without changing $\mathcal{R}_L$.

Let the set I^* contain exactly one ad per advertiser, arbitrarily chosen among the best ads for this advertiser. The inner maximization problem can then be simplified as:

$$\max_{i_1,\ldots,i_N\in I^*} \mathcal{R}_L(i_1,\ldots,i_N) \;=\; \sum_{p\in L} \gamma_p\, b_{i_p}\, \beta_{i_p}(x)$$

where all the indices $i_1,\ldots,i_N$ are distinct, and subject to the reserve constraints.

Assume that this maximization problem has a solution $i_1,\ldots,i_N$, meaning that there is a feasible ad placement solution for the layout L. For $k = 1\ldots N$, let us define $I_k^* \subset I^*$ as

$$I_k^* = \operatorname*{Arg\,Max}_{i\in I^*\setminus\{i_1,\ldots,i_{k-1}\}} b_i\beta_i(x)\,.$$

It is easy to see that I_k^* intersects $\{i_k,\ldots,i_N\}$ because, were it not the case, replacing i_k by any element of I_k^* would increase $\mathcal{R}_L$ without violating any of the constraints. Furthermore it is easy to see that $i_k \in I_k^*$ because, were it not the case, there would be $h > k$ such that $i_h \in I_k^*$, and swapping i_k and i_h would increase $\mathcal{R}_L$ without violating any of the constraints.

Therefore, if the inner maximization problem admits a solution, we can compute a solution by recursively picking $i_1,\ldots,i_N$ from $I_1^*, I_2^*, \ldots, I_N^*$. This can be done efficiently by first sorting the $b_i\beta_i(x)$ in decreasing order, and then greedily assigning ads to the best positions subject to the reserve constraints. This operation has to be repeated for all possible layouts, including of course the empty layout.

The same analysis can be carried out for click prediction estimates expressed as arbitrary monotone combination of a position term $\gamma_p(x)$ and an ad term $\beta_i(x)$, as shown, for instance, by Graepel et al. (2010).

Appendix B. Confidence Intervals

Section 4.4 explains how to obtain improved confidence intervals by replacing the unbiased importance sampling estimator (7) by the clipped importance sampling estimator (10). This appendix provides details that could have obscured the main message.

B.1 Outer Confidence Interval

We first address the computation of the outer confidence interval (12) which describes how the estimator $\widehat{Y}^*$ approaches the clipped expectation $\bar{Y}^*$.

$$\bar{Y}^* = \int_\omega \ell(\omega)\,\bar{w}(\omega)\,\mathrm{P}(\omega) \;\approx\; \widehat{Y}^* = \frac{1}{n}\sum_{i=1}^{n} \ell(\omega_i)\,\bar{w}(\omega_i)\,.$$

Since the samples $\ell(\omega_i)\,\bar{w}(\omega_i)$ are independent and identically distributed, the central limit theorem (e.g., Cramér, 1946, Section 17.4) states that the empirical average $\widehat{Y}^*$ converges in law to a normal distribution of mean $\bar{Y}^* = \mathbb{E}[\ell(\omega)\,\bar{w}(\omega)]$ and variance $\bar{V} = \mathrm{var}[\ell(\omega)\,\bar{w}(\omega)]$. Since this convergence usually occurs quickly, it is widely accepted to write

$$\mathbb{P}\Big\{\, \widehat{Y}^* - \varepsilon_R \le \bar{Y}^* \le \widehat{Y}^* + \varepsilon_R \,\Big\} \ge 1-\delta,$$

with

$$\varepsilon_R \;=\; \mathrm{erf}^{-1}(1-\delta)\,\sqrt{2\bar{V}}\,. \tag{25}$$

and to estimate the variance $\bar{V}$ using the sample variance $\widehat{V}$

$$\bar{V} \;\approx\; \widehat{V} = \frac{1}{n-1}\sum_{i=1}^{n}\Big(\ell(\omega_i)\,\bar{w}(\omega_i) - \widehat{Y}^*\Big)^2.$$

This approach works well when the ratio ceiling R is relatively small. However the presence of a few very large ratios makes the variance estimation noisy and might slow down the central limit convergence.

The first remedy is to bound the variance more rigorously. For instance, the following bound results from (Maurer and Pontil, 2009, Theorem 10).

$$\mathbb{P}\left\{ \sqrt{\bar{V}} \;>\; \sqrt{\widehat{V}} \;+\; (M-m)R\sqrt{\frac{2\log(2/\delta)}{n-1}} \right\} \le \delta$$

Combining this bound with (25) gives a confidence interval valid with probability greater than $1-2\delta$. Although this approach eliminates the potential problems related to the variance estimation, it does not address the potentially slow convergence of the central limit theorem.

The next remedy is to rely on *empirical Bernstein bounds* to derive rigorous confidence intervals that leverage both the sample mean and the sample variance (Audibert et al., 2007; Maurer and Pontil, 2009).

Theorem 1 (Empirical Bernstein bound) (Maurer and Pontil, 2009, thm 4)
Let $X, X_1, X_2, \ldots, X_n$ be i.i.d. random variable with values in $[a,b]$ and let $\delta > 0$. Then, with probability at least $1-\delta$,

$$\mathbb{E}[X] - M_n \;\le\; \sqrt{\frac{2V_n\log(2/\delta)}{n}} + (b-a)\,\frac{7\log(2/\delta)}{3(n-1)}\,,$$

where M_n and V_n respectively are the sample mean and variance

$$M_n = \frac{1}{n}\sum_{i=1}^{n} X_i\,, \qquad V_n = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - M_n)^2\,.$$

Applying this theorem to both $\ell(\omega_i)\,\bar{w}(\omega_i)$ and $-\ell(\omega_i)\,\bar{w}(\omega_i)$ provides confidence intervals that hold for for the worst possible distribution of the variables $\ell(\omega)$ and $\bar{w}(\omega)$.

$$\mathbb{P}\Big\{\ \widehat{Y}^* - \varepsilon_R \le \bar{Y}^* \le \widehat{Y}^* + \varepsilon_R\ \Big\} \ge 1 - 2\delta$$

where

$$\varepsilon_R \;=\; \sqrt{\frac{2\widehat{V}\log(2/\delta)}{n}} + M R \frac{7\log(2/\delta)}{3(n-1)}. \tag{26}$$

Because they hold for the worst possible distribution, confidence intervals obtained in this way are less tight than confidence intervals based on the central limit theorem. On the other hand, thanks to the Bernstein bound, they remains reasonably competitive, and they provide a much stronger guarantee.

B.2 Inner Confidence Interval

Inner confidence intervals are derived from inequality (14) which bounds the difference between the counterfactual expectation Y^* and the clipped expectation $\bar{Y}^*$:

$$0 \;\le\; Y^* - \bar{Y}^* \;\le\; M\,(1 - \bar{W}^*)\,.$$

The constant M is defined by assumption (8). The first step of the derivation consists in obtaining a lower bound of $\bar{W}^* - \widehat{W}^*$ using either the central limit theorem or an empirical Bernstein bound.

For instance, applying theorem 1 to $-\bar{w}(\omega_i)$ yields

$$\mathbb{P}\left\{\ \bar{W}^* \;\ge \widehat{W}^* - \sqrt{\frac{2\widehat{V}_w\log(2/\delta)}{n}} - R\frac{7\log(2/\delta)}{3(n-1)}\ \right\} \;\ge\; 1-\delta$$

where $\widehat{V}_w$ is the sample variance of the clipped weights

$$\widehat{V}_w \;=\; \frac{1}{n-1}\sum_{i=1}^{n}\Big(\bar{w}(\omega_i) - \widehat{W}^*\Big)^2.$$

Replacing in inequality (14) gives the outer confidence interval

$$\mathbb{P}\Big\{\ \bar{Y}^* \;\le\; Y^* \;\le\; \bar{Y}^* + M(1 - \widehat{W}^* + \xi_R)\ \Big\} \ge 1 - \delta\,.$$

with

$$\xi_R \;=\; \sqrt{\frac{2\widehat{V}_w\log(2/\delta)}{n}} + R\frac{7\log(2/\delta)}{3(n-1)}\;. \tag{27}$$

Note that $1 - \widehat{W} + \xi_R$ can occasionally be negative. This occurs in the unlucky cases where the confidence interval is violated, with probability smaller than δ.

Putting together the inner and outer confidence intervals,

$$\mathbb{P}\Big\{\ \widehat{Y}^* - \varepsilon_R \le Y^* \le \widehat{Y}^* + M(1 - \widehat{W}^* + \xi_R) + \varepsilon_R\ \Big\} \ge 1 - 3\delta\,,$$

with ε_R and ξ_R computed as described in expressions (26) and (27).

Appendix C. Counterfactual Differences

We now seek to estimate the difference $Y^+ - Y^*$ of the expectations of a same quantity $\ell(\omega)$ under two different counterfactual distributions $\mathrm{P}^+(\omega)$ and $\mathrm{P}^*(\omega)$. These expectations are often affected by variables whose value is left unchanged by the interventions under consideration. For instance, seasonal effects can have very large effects on the number of ad clicks. When these variables affect both Y^+ and Y^* in similar ways, we can obtain substantially better confidence intervals for the difference $Y^+ - Y^*$.

In addition to the notation ω representing all the variables in the structural equation model, we use notation υ to represent all the variables that are not direct or indirect effects of variables affected by the interventions under consideration.

Let $\zeta(\upsilon)$ be a known function believed to be a good predictor of the quantity $\ell(\omega)$ whose counterfactual expectation is sought. Since $\mathrm{P}^*(\upsilon) = \mathrm{P}(\upsilon)$, the following equality holds regardless of the quality of this prediction:

$$\begin{aligned} Y^* \;=\; \int_\omega \ell(\omega)\,\mathrm{P}^*(\omega) \;&=\; \int_\upsilon \zeta(\upsilon)\,\mathrm{P}^*(\upsilon) \;+\; \int_\omega \big[\ell(\omega)-\zeta(\upsilon)\big]\,\mathrm{P}^*(\omega) \\ &=\; \int_\upsilon \zeta(\upsilon)\,\mathrm{P}(\upsilon) \;+\; \int_\omega \big[\ell(\omega)-\zeta(\upsilon)\big]\; w(\omega)\,\mathrm{P}(\omega)\;. \end{aligned} \tag{28}$$

Decomposing both Y^+ and Y^* in this way and computing the difference,

$$Y^+ - Y^* \;=\; \int_\omega \; [\ell(\omega)-\zeta(\upsilon)]\,\Delta w(\omega)\,\mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^{n} \big[\,\ell(\omega_i)-\zeta(\upsilon_i)\,\big]\,\Delta w(\omega_i)\,,$$

$$\text{with} \qquad \Delta w(\omega) \;=\; \frac{\mathrm{P}^+(\omega)}{\mathrm{P}(\omega)} - \frac{\mathrm{P}^*(\omega)}{\mathrm{P}(\omega)} \;=\; \frac{\mathrm{P}^+(\omega)-\mathrm{P}^*(\omega)}{\mathrm{P}(\omega)}\,.$$

The outer confidence interval size is reduced if the variance of the residual $\ell(\omega)-\zeta(\upsilon)$ is smaller than the variance of the original variable $\ell(\omega)$. For instance, a suitable predictor function $\zeta(\upsilon)$ can significantly capture the seasonal click yield variations regardless of the interventions under consideration. Even a constant predictor function can considerably change the variance of the outer confidence interval. Therefore, in the absence of better predictor, we still can (and always should) center the integrand using a constant predictor.

The rest of this appendix describes how to construct confidence intervals for the estimation of counterfactual differences. Additional bookkeeping is required because both the weights $\Delta w(\omega_i)$ and the integrand $\ell(\omega)-\zeta(\upsilon)$ can be positive or negative. We use the notation υ to represent the variables of the structural equation model that are left unchanged by the intervention under considerations. Such variables satisfy the relations $\mathrm{P}^*(\upsilon) = \mathrm{P}(\upsilon)$ and $\mathrm{P}^*(\omega) = \mathrm{P}^*(\,\omega\backslash\upsilon\,|\upsilon)\,\mathrm{P}(\upsilon)$, where we use notation $\omega\backslash\upsilon$ to denote all remaining variables in the structural equation model. An invariant predictor is then a function $\zeta(\upsilon)$ that is believed to be a good predictor of $\ell(\omega)$. In particular, it is expected that $\mathrm{var}[\ell(\omega)-\zeta(\upsilon)]$ is smaller than $\mathrm{var}[\ell(\omega)]$.

C.1 Inner Confidence Interval with Dependent Bounds

We first describe how to construct finer inner confidence intervals by using more refined bounds on $\ell(\omega)$. In particular, instead of the simple bound (8), we can use bounds that depend on invariant variables:

$$\forall\omega \qquad m \le m(\upsilon) \le \ell(\omega) \le M(\upsilon) \le M\,.$$

The key observation is the equality

$$\mathbb{E}[w^*(\omega)|\upsilon] \;=\; \int_{\omega\backslash\upsilon} w^*(\omega)\,\mathrm{P}(\,\omega\backslash\upsilon\,|\upsilon) \;=\; \int_{\omega\backslash\upsilon} \frac{P^*(\,\omega\backslash\upsilon\,|\upsilon)\,P(\upsilon)}{P(\,\omega\backslash\upsilon\,|\upsilon)\,P(\upsilon)}\,P(\,\omega\backslash\upsilon\,|\upsilon) \;=\; 1\,.$$

We can then write

$$\begin{aligned} Y^* - \bar{Y}^* \;&=\; \int_\omega \big[w^*(\omega) - \bar{w}^*(\omega)\big]\;\ell(\omega)\;\mathrm{P}(\omega) \;\le\; \int_\upsilon \mathbb{E}\big[\,w^*(\omega) - \bar{w}^*(\omega)\,|\,\upsilon\,\big]\,M(\upsilon)\;\mathrm{P}(\upsilon) \\ &=\; \int_\upsilon (\,1 - \mathbb{E}[\bar{w}^*(\omega)|\upsilon]\,)\,M(\upsilon)\;\mathrm{P}(\upsilon) \;=\; \int_\omega (\,1-\bar{w}^*(\omega)\,)\,M(\upsilon)\;\mathrm{P}(\omega) \;=\; \mathcal{B}_{\mathrm{hi}}\,. \end{aligned}$$

Using a similar derivation for the lower bound $\mathcal{B}_{\mathrm{lo}}$, we obtain the inequality

$$\mathcal{B}_{\mathrm{lo}} \;\le\; Y^* - \bar{Y}^* \;\le\; \mathcal{B}_{\mathrm{hi}}$$

With the notations

$$\widehat{\mathcal{B}}_{\mathrm{lo}} = \frac{1}{n}\sum_{i=1}^n (1-\bar{w}^*(\omega_i))\,m(\upsilon_i)\,, \qquad \widehat{\mathcal{B}}_{\mathrm{hi}} = \frac{1}{n}\sum_{i=1}^n (1-\bar{w}^*(\omega_i))\,M(\upsilon_i)\,,$$

$$\widehat{V}_{\mathrm{lo}} = \frac{1}{n-1}\sum_{i=1}^n \Big[\,(1-\bar{w}^*(\omega_i))\,m(\upsilon_i) - \widehat{\mathcal{B}}_{\mathrm{lo}}\,\Big]^2\,, \quad \widehat{V}_{\mathrm{hi}} = \frac{1}{n-1}\sum_{i=1}^n \Big[\,(1-\bar{w}^*(\omega_i))\,M(\upsilon_i) - \widehat{\mathcal{B}}_{\mathrm{hi}}\,\Big]^2\,,$$

$$\xi_{\mathrm{lo}} \;=\; \sqrt{\frac{2\,\widehat{V}_{\mathrm{lo}}\log(2/\delta)}{n}} + |m|R\,\frac{7\log(2/\delta)}{3(n-1)}\,, \quad \xi_{\mathrm{hi}} \;=\; \sqrt{\frac{2\,\widehat{V}_{\mathrm{hi}}\log(2/\delta)}{n}} + |M|R\,\frac{7\log(2/\delta)}{3(n-1)}\,,$$

two applications of theorem 1 give the inner confidence interval:

$$\mathbb{P}\Big\{\;\bar{Y}^* + \widehat{\mathcal{B}}_{\mathrm{lo}} - \xi_{\mathrm{lo}} \;\le\; Y^* \;\le\; \bar{Y}^* + \widehat{\mathcal{B}}_{\mathrm{hi}} + \xi_{\mathrm{hi}}\;\Big\} \;\ge\; 1-2\delta\,.$$

C.2 Confidence Intervals for Counterfactual Differences

We now describe how to leverage invariant predictors in order to construct tighter confidence intervals for the difference of two counterfactual expectations.

$$Y^+ - Y^* \;\approx\; \frac{1}{n}\sum_{i=1}^n \big[\,\ell(\omega_i) - \zeta(\upsilon_i)\,\big]\,\Delta w(\omega_i) \qquad \text{with}\;\; \Delta w(\omega) = \frac{\mathrm{P}^+(\omega) - \mathrm{P}^*(\omega)}{\mathrm{P}(\omega)}\,.$$

Let us define the reweigthing ratios $w^+(\omega) = \mathrm{P}^+(\omega)/\mathrm{P}(\omega)$ and $w^*(\omega) = \mathrm{P}^*(\omega)/\mathrm{P}(\omega)$, their clipped variants $\bar{w}^+(\omega)$ and $\bar{w}^*(\omega)$, and the clipped centered expectations

$$\bar{Y}_c^+ = \int_\omega [\ell(\omega) - \zeta(\upsilon)]\,\bar{w}^+(\omega)\,\mathrm{P}(\omega) \quad \text{and} \quad \bar{Y}_c^* = \int_\omega [\ell(\omega) - \zeta(\upsilon)]\,\bar{w}^*(\omega)\,\mathrm{P}(\omega)\,.$$

The outer confidence interval is obtained by applying the techniques of Section B.1 to

$$\bar{Y}_c^+ - \bar{Y}_c^* \;=\; \int_\omega \,[\,\ell(\omega) - \zeta(\upsilon)\,]\,[\,\bar{w}^+(\omega) - \bar{w}^*(\omega)\,]\;\mathrm{P}(\omega)\,.$$

Since the weights $\bar{w}^+ - \bar{w}^*$ can be positive or negative, adding or removing a constant to $\ell(\omega)$ can considerably change the variance of the outer confidence interval. This means that one should

always use a predictor. Even a *constant predictor* can vastly improve the outer confidence interval difference.

The inner confidence interval is then obtained by writing the difference

$$\begin{aligned}\big(Y^+ - Y^*\big) - \big(\bar{Y}_c^+ - \bar{Y}_c^*\big) &= \int_\omega \big[\,\ell(\omega) - \zeta(\upsilon)\,\big]\ \big[w^+(\omega) - \bar{w}^+(\omega)\big]\,\mathrm{P}(\omega) \\ &- \int_\omega \big[\,\ell(\omega) - \zeta(\upsilon)\,\big]\ \big[w^*(\omega) - \bar{w}^*(\omega)\big]\,\mathrm{P}(\omega)\end{aligned}$$

and bounding both terms by leveraging υ-dependent bounds on the integrand:

$$\forall \omega \qquad -M \;\le\; -\zeta(\upsilon) \;\le\; \ell(\omega) - \zeta(\upsilon) \;\le\; M - \zeta(\upsilon) \;\le\; M\,.$$

This can be achieved as shown in Section C.1.

Appendix D. Counterfactual Derivatives

We now consider interventions that depend on a continuous parameter θ. For instance, we might want to know what the performance of the ad placement engine would have been if we had used a parametrized scoring model. Let $\mathrm{P}^\theta(\omega)$ represent the counterfactual Markov factorization associated with this intervention. Let Y^θ be the counterfactual expectation of $\ell(\omega)$ under distribution P^θ.

Computing the derivative of (28) immediately gives

$$\frac{\partial Y^\theta}{\partial \theta} \;=\; \int_w \big[\,\ell(\omega) - \zeta(\upsilon)\,\big]\, w'_\theta(\omega)\,\mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^n \big[\,\ell(\omega_i) - \zeta(\upsilon_i)\,\big]\, w'_\theta(\omega_i)$$

$$\text{with}\quad w_\theta(\omega) = \frac{\mathrm{P}^\theta(\omega)}{\mathrm{P}(\omega)} \qquad \text{and}\quad w'_\theta(\omega) = \frac{\partial w_\theta(\omega)}{\partial \theta} = w_\theta(\omega)\,\frac{\partial \log \mathrm{P}^\theta(\omega)}{\partial \theta}\,. \tag{29}$$

Replacing the expressions $\mathrm{P}(\omega)$ and $\mathrm{P}^\theta(\omega)$ by the corresponding Markov factorizations gives many opportunities to simplify the reweighting ratio $w'_\theta(\omega)$. The term $w_\theta(\omega)$ simplifies as shown in (6). The derivative of $\log \mathrm{P}^\theta(\omega)$ depends only on the factors parametrized by θ. Therefore, in order to evaluate $w'_\theta(\omega)$, we only need to know the few factors affected by the intervention.

Higher order derivatives can be estimated using the same approach. For instance,

$$\frac{\partial^2 Y^\theta}{\partial \theta_i \partial \theta_j} \;=\; \int_w \big[\,\ell(\omega) - \zeta(\upsilon)\,\big]\, w''_{ij}(\omega)\,\mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^n \big[\,\ell(\omega_i) - \zeta(\upsilon_i)\,\big]\, w''_{ij}(\omega_i)$$

$$\text{with}\quad w''_{ij}(\omega) = \frac{\partial^2 w_\theta(\omega)}{\partial \theta_i \partial \theta_j} = w_\theta(\omega)\,\frac{\partial \log \mathrm{P}^\theta(\omega)}{\partial \theta_i}\,\frac{\partial \log \mathrm{P}^\theta(\omega)}{\partial \theta_j} + w_\theta(\omega)\,\frac{\partial^2 \log \mathrm{P}^\theta(\omega)}{\partial \theta_i \partial \theta_j}\,.$$

The second term in $w''_{ij}(\omega)$ vanishes when θ_i and θ_j parametrize distinct factors in $\mathrm{P}^\theta(\omega)$.

D.1 Infinitesimal Interventions and Policy Gradient

Expression (29) becomes particularly attractive when $\mathrm{P}(\omega) = \mathrm{P}^\theta(\omega)$, that is, when one seeks derivatives that describe the effect of an infinitesimal intervention on the system from which the data was collected. The resulting expression is then identical to the celebrated *policy gradient* (Aleksandrov et al., 1968; Glynn, 1987; Williams, 1992) which expresses how the accumulated rewards

in a reinforcement learning problem are affected by small changes of the parameters of the policy function.

$$\frac{\partial Y^\theta}{\partial \theta} \;=\; \int_\omega \big[\,\ell(\omega)-\zeta(\upsilon)\,\big]\, w'_\theta(\omega)\, \mathrm{P}^\theta(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^{n} \big[\,\ell(\omega_i)-\zeta(\upsilon_i)\,\big]\, w'_\theta(\omega_i)$$

$$\text{where } \omega_i \text{ are sampled i.i.d. from } \mathrm{P}^\theta \text{ and } w'_\theta(\omega) \;=\; \frac{\partial \log \mathrm{P}^\theta(\omega)}{\partial \theta}\,.$$

Sampling from $\mathrm{P}^\theta(\omega)$ eliminates the potentially large ratio $w_\theta(\omega)$ that usually plagues importance sampling approaches. Choosing a parametrized distribution that depends smoothly on θ is then sufficient to contain the size of the weights $w'_\theta(\omega)$. Since the weights can be positive or negative, centering the integrand with a prediction function $\zeta(\upsilon)$ remains very important. Even a constant predictor ζ can substantially reduce the variance

$$\begin{aligned}\mathrm{var}[\,(\ell(\omega)-\zeta)\,w'_\theta(\omega)\,] \;&=\; \mathrm{var}[\,\ell(\omega)\,w'_\theta(\omega)-\zeta\,w'_\theta(\omega)\,] \\ &=\; \mathrm{var}[\ell(\omega)\,w'_\theta(\omega)] - 2\,\zeta\,\mathrm{cov}[\,\ell(\omega)\,w'_\theta(\omega),\, w'_\theta(\omega)\,] + \zeta^2\,\mathrm{var}[w'_\theta(\omega)]\end{aligned}$$

whose minimum is reached for $\zeta = \dfrac{\mathrm{cov}[\ell(\omega)w'_\theta(\omega), w'_\theta(\omega)]}{\mathrm{var}[w'_\theta(\omega)]} = \dfrac{\mathbb{E}[\ell(\omega)w'_\theta(\omega)^2]}{\mathbb{E}[w'_\theta(\omega)^2]}$.

We sometimes want to evaluate expectations under a counterfactual distribution that is too far from the actual distribution to obtain reasonable confidence intervals. Suppose, for instance, that we are unable to reliably estimate which click yield would have been observed if we had used a certain parameter θ^* for the scoring models. We still can estimate how quickly and in which direction the click yield would have changed if we had slightly moved the current scoring model parameters θ in the direction of the target θ^*. Although such an answer is not as good as a reliable estimate of Y^{θ^*}, it is certainly better than no answer.

D.2 Off-Policy Gradient

We assume in this subsection that the parametrized probability distribution $\mathrm{P}^\theta(\omega)$ is regular enough to ensure that all the derivatives of interest are defined and that the event $\{w_\theta(\omega)=R\}$ has probability zero. Furthermore, in order to simplify the exposition, the following derivation does not leverage an invariant predictor function.

Estimating derivatives using data sampled from a distribution $\mathrm{P}(\omega)$ different from $\mathrm{P}^\theta(\omega)$ is more challenging because the ratios $w_\theta(\omega_i)$ in Equation (29) can take very large values. However it is comparatively easy to estimate the derivatives of lower and upper bounds using a slightly different way to clip the weights. Using notation $\mathbb{1}(x)$ represent the indicator function, equal to one if condition x is true and zero otherwise, let us define respectively the clipped weights $\bar{w}^{\mathrm{Z}}_\theta$ and the capped weights $\bar{w}^{\mathrm{M}}_\theta$:

$$\bar{w}^{\mathrm{Z}}_\theta(\omega) = w_\theta(\omega)\,\mathbb{1}\{\mathrm{P}^*(\omega) < R\,\mathrm{P}(\omega)\} \quad \text{and} \quad \bar{w}^{\mathrm{M}}_\theta(\omega) = \min\{w_\theta(\omega), R\}\,.$$

Although Section 4.4 illustrates the use of clipped weights, the confidence interval derivation can be easily extended to the capped weights. Defining the capped quantities

$$\bar{Y}^\theta = \int_\omega \ell(\omega)\,\bar{w}^{\mathrm{M}}_\theta(\omega)\,\mathrm{P}(\omega) \quad \text{and} \quad \bar{W}^\theta = \int_\omega \bar{w}^{\mathrm{M}}_\theta(\omega)\,\mathrm{P}(\omega)$$

and writing

$$\begin{aligned} 0 \;\le\; Y^\theta - \bar{Y}^\theta \;&=\; \int_{\omega\in\Omega\setminus\Omega_R} \ell(\omega)\,(\,\mathrm{P}^*(\omega) - R\,\mathrm{P}(\omega)\,) \\ &\le\; M\left(1-\mathrm{P}^*(\Omega_R) - R\,\mathrm{P}(\Omega\setminus\Omega_R)\right) \;=\; M\left(1-\int_\omega \bar{w}_\theta^{\mathrm{M}}(\omega)\,\mathrm{P}(\omega)\right) \end{aligned}$$

yields the inequality

$$\bar{Y}^\theta \;\le\; Y^\theta \;\le\; \bar{Y}^\theta + M(1-\bar{W}^\theta)\,. \tag{30}$$

In order to obtain reliable estimates of the derivatives of these upper and lower bounds, it is of course sufficient to obtain reliable estimates of the derivatives of $\bar{Y}^\theta$ and $\bar{W}^\theta$. By separately considering the cases $w_\theta(\omega) < R$ and $w_\theta(\omega) > R$, we easily obtain the relation

$$\bar{w}_\theta^{\mathrm{M}'}(\omega) \;=\; \frac{\partial \bar{w}_\theta^{\mathrm{M}}(\omega)}{\partial\theta} \;=\; \bar{w}_\theta^{\mathrm{Z}}(\omega)\,\frac{\partial \log P^\theta(\omega)}{\partial\theta} \quad \text{when} \;\; w_\theta(\omega)\neq R$$

and, thanks to the regularity assumptions, we can write

$$\begin{aligned} \frac{\partial \bar{Y}^\theta}{\partial\theta} \;&=\; \int_\omega \ell(\omega)\,\bar{w}_\theta^{\mathrm{M}'}(\omega)\,\mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^n \ell(\omega_i)\,\bar{w}_\theta^{\mathrm{M}'}(\omega_i)\,, \\ \frac{\partial \bar{W}^\theta}{\partial\theta} \;&=\; \int_\omega \bar{w}_\theta^{\mathrm{M}'}(\omega)\,\mathrm{P}(\omega) \;\approx\; \frac{1}{n}\sum_{i=1}^n \bar{w}_\theta^{\mathrm{M}'}(\omega_i)\,, \end{aligned}$$

Estimating these derivatives is considerably easier than using approximation (29) because they involve the bounded quantity $\bar{w}_\theta^{\mathrm{Z}}(\omega)$ instead of the potentially large ratio $w_\theta(\omega)$. It is still necessary to choose a sufficiently smooth sampling distribution $\mathrm{P}(\omega)$ to limit the magnitude of $\partial\log\mathrm{P}^\theta/\partial\theta$.

Such derivatives are very useful to drive optimization algorithms. Assume for instance that we want to find the parameter θ that maximizes the counterfactual expectation Y^θ as illustrated in Section 6.3. Maximizing the estimate obtained using approximation (5) could reach its maximum for a value of θ that is poorly explored by the actual distribution. Maximizing an estimate of the lower bound (30) ensures that the optimization algorithm finds a trustworthy answer.

Appendix E. Uniform Empirical Bernstein Bounds

This appendix reviews the uniform empirical Bernstein bound given by Maurer and Pontil (2009) and describes how it can be used to construct the uniform confidence interval (21). The first step consists of characterizing the size of a family $\mathcal{F}$ of functions mapping a space $\mathcal{X}$ into the interval $[a,b]\subset\mathbb{R}$. Given n points $\mathbf{x}=(x_1\ldots x_n)\in\mathcal{X}^n$, the trace $\mathcal{F}(\mathbf{x})\in\mathbb{R}^n$ is the set of vectors $\big(f(x_1),\ldots,f(x_n)\big)$ for all functions $f\in\mathcal{F}$.

Definition 2 (Covering numbers, etc.) *Given $\varepsilon>0$, the covering number $\mathcal{N}(\mathbf{x},\varepsilon,\mathcal{F})$ is the smallest possible cardinality of a subset $C\subset\mathcal{F}(\mathbf{x})$ satisfying the condition*

$$\forall v\in\mathcal{F}(\mathbf{x}) \quad \exists c\in C \quad \max_{i=1\ldots n} |v_i-c_i| \le \varepsilon\,,$$

and the growth function $\mathcal{N}(n,\varepsilon,\mathcal{F})$ is

$$\mathcal{N}(n,\varepsilon,\mathcal{F}) \;=\; \sup_{\mathbf{x}\in\mathcal{X}^n} \mathcal{N}(\mathbf{x},\varepsilon,\mathcal{F})\,.$$

Thanks to a famous combinatorial lemma (Vapnik and Chervonenkis, 1968, 1971; Sauer, 1972), for many usual parametric families $\mathcal{F}$, the growth function $\mathcal{N}(n,\varepsilon,\mathcal{F})$ increases at most polynomially[13] with both n and $1/\varepsilon$.

Theorem 3 (Uniform empirical Bernstein bound) (Maurer and Pontil, 2009, thm 6)
Let $\delta \in (0,1)$, $n >= 16$. Let $X, X_1, \ldots, X_n$ be i.i.d. random variables with values in $\mathcal{X}$. Let $\mathcal{F}$ be a set of functions mapping $\mathcal{X}$ into $[a,b] \subset \mathbb{R}$ and let $\mathcal{M}(n) = 10\,\mathcal{N}(2n,\mathcal{F},1/n)$. Then we probability at least $1-\delta$,

$$\forall f \in \mathcal{F},\ \ \mathbb{E}[f(X)] - M_n \ \leq\ \sqrt{\frac{18\,V_n \log(\mathcal{M}(n)/\delta)}{n}} + (b-a)\frac{15\log(\mathcal{M}(n)/\delta)}{n-1}\ ,$$

where M_n and V_n respectively are the sample mean and variance

$$M_n = \frac{1}{n}\sum_{i=1}^{n} f(X_i)\ , \qquad V_n = \frac{1}{n-1}\sum_{i=1}^{n}(f(X_i)-M_n)^2\ .$$

The statement of this theorem emphasizes its similarity with the non-uniform empirical Bernstein bound (theorem 1). Although the constants are less attractive, the uniform bound still converges to zero when n increases, provided of course that $\mathcal{M}(n) = 10\,\mathcal{N}(2n,\mathcal{F},1/n)$ grows polynomially with n.

Let us then define the family of functions

$$\mathcal{F} = \{\ f_\theta : \omega \mapsto \ell(\omega)\bar{w}_\theta^{\mathrm{M}}(\omega)\ ,\ \ g_\theta : \omega \mapsto \bar{w}_\theta^{\mathrm{M}}(\omega)\ ,\ \ \forall \theta \in \mathcal{F}\ \}\ ,$$

and use the uniform empirical Bernstein bound to derive an outer inequality similar to (26) and an inner inequality similar to (27). The theorem implies that, with probability $1-\delta$, both inequalities are simultaneously true for all values of the parameter θ. The uniform confidence interval (21) then follows directly.

References

V M. Aleksandrov, V. I. Sysoyev, and V. V. Shemeneva. Stochastic optimization. *Engineering Cybernetics*, 5:11–16, 1968.

Susan Athey and Denis Nekipelov. A structural model of sponsored search advertising. Working paper, 2010. URL `http://kuznets.harvard.edu/~athey/papers/Structural_Sponsored_Search.pdf`.

Jean-Yves Audibert, Remi Munos, and Csaba Szepesvári. Tuning bandit algorithms in stochastic environments. In *Proc. 18th International Conference on Algorithmic Learning Theory (ALT 2007)*, pages 150–165, 2007.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fisher. Finite time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2–3):235–256, 2002.

13. For a simple proof of this fact, slice $[a,b]$ into intervals S_k of maximal width ε and apply the lemma to the family of indicator functions $(x_i,S_k) \mapsto \mathbb{1}\{f(x_i) \in S_k\}$.

Heejung Bang and James M. Robins. Doubly robust estimation in missing data and causal inference models. *Biometrics*, 61:692–972, 2005.

Dirk Bergemann and Maher Said. Dynamic auctions: a survey. Discussion Paper 1757R, Cowles Foundation for Research in Economics, Yale University, 2010.

Léon Bottou. From machine learning to machine reasoning. `http://arxiv.org/abs/1102.1808v3`, Feb 2011.

Leo Breiman. Statistical modeling: The two cultures. *Statistical Science*, 16(3):199–231, 2001.

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in Neural Information Processing Systems 24*, pages 2249–2257. NIPS Foundation, 2011.

C. R. Charig, D. R. Webb, S. R. Payne, and J. E. A. Wickham. Comparison of treatment of renal calculi by open surgery, percutaneous nephrolithotomy, and extracorporeal shockwave lithotripsy. *British Medical Journal (Clin Res Ed)*, 292(6254):879–882, 1986.

Denis X. Charles and D. Max Chickering. Optimization for paid search auctions. Manuscript in preparation, 2012.

Denis X. Charles, D. Max Chickering, and Patrice Simard. Micro-market experimentation for paid search. Manuscript in preparation, 2012.

Harald Cramér. *Mathematical Methods of Statistics*. Princeton University Press, 1946.

Denver Dash. *Caveats for Causal Reasoning with Equilibrium Models*. PhD thesis, University of Pittsburgh, 2003.

Miroslav Dudík, Dimitru Erhan, John Langford, and Lihong Li. Sample-efficient nonstationary-policy evaluation for contextual bandits. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, pages 247–254, 2012.

Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97(1):242–259, 2007.

Alan Genz. Numerical computation of multivariate normal probabilities. *Journal Computation of Multivariate Normal Probabilities*, 1:141–149, 1992.

John C. Gittins. *Bandit Processes and Dynamic Allocation Indices*. Wiley, 1989.

Peter W. Glynn. Likelihood ratio gradient estimation: an overview. In *Proceedings of the 1987 Winter Simulation Conference*, pages 366–375, 1987.

John Goodwin. Microsoft adCenter. Personal communication, 2011.

Thore Graepel, Joaquin Quinonero Candela, Thomas Borchert, and Ralf Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *Proceedings of the 27th International Conference on Machine Learning (ICML 2010), Invited Applications Track*. Omnipress, 2010.

Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. Controlled experiments on the web: Survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1):140–181, July 2008.

Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems, October 2010. http://www.cs.mcgill.ca/~vkules/bandits.pdf.

Sébastien Lahaie and R. Preston McAfee. Efficient ranking in sponsored search. In *Proc. 7th International Workshop on Internet and Network Economics (WINE 2011)*, pages 254–265. LNCS 7090, Springer, 2011.

Lev Landau and Evgeny Lifshitz. *Course in Theoretical Physics, Volume 1: Mechanics*. Pergamon Press, 1969. 2nd edition.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in Neural Information Processing Systems 20*, pages 817–824. MIT Press, Cambridge, MA, 2008.

Steffen L. Lauritzen and Thomas S. Richardson. Chain graph models and their causal interpretation. *Journal of the Royal Statistical Society, Series B*, 64:321–361, 2002.

David K. Lewis. *Counterfactuals*. Harvard University Press, 1973. 2nd edition: Wiley-Blackwell, 2001.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on the World Wide Web (WWW 2010)*, pages 661–670. ACM, 2010.

Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proc. 4th ACM International Conference on Web Search and Data Mining (WSDM 2011)*, pages 297–306, 2011.

Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample-variance penalization. In *Proc. The 22nd Conference on Learning Theory (COLT 2009)*, 2009.

Paul Milgrom. *Putting Auction Theory to Work*. Cambridge University Press, 2004.

Roger B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000. 2nd edition: 2009.

Judea Pearl. Causal inference in statistics: an overview. *Statistics Surveys*, 3:96–146, 2009.

Judea Pearl. The do-calculus revisited. In *Proc. Twenty-Eighth Conference on Uncertainty in Artificial Intelligence (UAI-2012)*, pages 3–11, 2012.

Linda E. Reichl. *A Modern Course in Statistical Physics, 2nd Edition*. Wiley, 1998.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

James M. Robins, Miguel Angel Hernan, and Babette Brumback. Marginal structural models and causal inference in epidemiology. *Epidemiology*, 11(5):550–560, Sep 2000.

Norbert Sauer. On the density of families of sets. *Journal of Combinatorial Theory*, 13:145–147, 1972.

Yevgeny Seldin, cois Laviolette Fran Nicolò Cesa-Bianchi, John Shawe-Taylor, and Peter Auer. PAC-Bayesian inequalities for martingales. *IEEE Transactions on Information Theory*, 58(12): 7086–7093, 2012.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

Edward H. Simpson. The interpretation of interaction in contingency tables. *Journal of the Royal Statistical Society, Ser. B*, 13:238–241, 1951.

Alexsandrs Slivkins. Contextual bandits with similarity information. *JMLR Conference and Workshop Proceedings*, 19:679–702, 2011.

Peter Spirtes and Richard Scheines. Causal inference of ambiguous manipulations. *Philosophy of Science*, 71(5):833–845, Dec 2004.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction and Search*. Springer Verlag, New York, 1993. 2nd edition: MIT Press, Cambridge (Mass.), 2011.

Stephen M. Stigler. A historical view of statistical concepts in psychology and educational research. *American Journal of Education*, 101(1):60–70, Nov 1992.

Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:985–1005, 2007.

Rich S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

Diane Tang, Ashish Agarwal, Deirdre O'Brien, and Mike Meyer. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings 16th Conference on Knowledge Discovery and Data Mining (KDD 2010)*, pages 17–26, 2010.

Vladimir N. Vapnik. *Estimation of Dependences based on Empirical Data*. Springer Series in Statistics. Springer Verlag, Berlin, New York, 1982.

Vladimir N. Vapnik and Alexey Ya. Chervonenkis. Uniform convegence of the frequencies of occurence of events to their probabilities. *Proc. Academy of Sciences of the USSR*, 181(4), 1968. English translation: *Soviet Mathematics - Doklady*, 9:915-918, 1968.

Vladimir N. Vapnik and Alexey Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

Hal R. Varian. Position auctions. *International Journal of Industrial Organization*, 25:1163–1178, 2007.

Hal R. Varian. Online ad auctions. *American Economic Review*, 99(2):430–434, 2009.

Joannes Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *Proc. European Conference on Machine Learning*, pages 437–448, 2005.

Georg H. von Wright. *Explanation and Understanding*. Cornell University Press, 1971.

Abraham Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.

Norbert Wiener. *Cybernetics, or Control and Communication in the Animal and the Machine*. Hermann et Cie (Paris), MIT Press (Cambridge, Mass.), Wiley and Sons (New York), 1948. 2nd Edition (expanded): MIT Press, Wiley and Sons, 1961.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(229–256), 1992.

James Woodward. *Making Things Happen*. Oxford University Press, 2005.

Sewall S. Wright. Correlation and causation. *Journal of Agricultural Research*, 20:557–585, 1921.

Journal of Machine Learning Research 14 (2013) 3261-3294 Submitted 4/12; Revised 1/13; Published 11/13

Multivariate Convex Regression with Adaptive Partitioning

Lauren A. Hannah LAH2178@COLUMBIA.EDU
Department of Statistics
Columbia University
New York, NY 10027, USA

David B. Dunson DUNSON@STAT.DUKE.EDU
Department of Statistical Science
Duke University
Durham, NC 27708, USA

Editor: Hui Zou

Abstract

We propose a new, nonparametric method for multivariate regression subject to convexity or concavity constraints on the response function. Convexity constraints are common in economics, statistics, operations research, financial engineering and optimization, but there is currently no multivariate method that is stable and computationally feasible for more than a few thousand observations. We introduce convex adaptive partitioning (CAP), which creates a globally convex regression model from locally linear estimates fit on adaptively selected covariate partitions. CAP is a computationally efficient, consistent method for convex regression. We demonstrate empirical performance by comparing the performance of CAP to other shape-constrained and unconstrained regression methods for predicting weekly wages and value function approximation for pricing American basket options.

Keywords: adaptive partitioning, convex regression, nonparametric regression, shape constraint, treed linear model

1. Introduction

Consider the regression model for $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^p$ and $y \in \mathbb{R}$,

$$y = f_0(\mathbf{x}) + \varepsilon,$$

where $f_0 : \mathbb{R}^p \to \mathbb{R}$ and ε is a mean 0 random variable. In this paper, we study the situation where f_0 is convex. That is,

$$\lambda f_0(\mathbf{x}_1) + (1-\lambda) f_0(\mathbf{x}_2) \geq f_0(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2),$$

for every $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$ and $\lambda \in (0,1)$. Given the observations $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, we would like to estimate f_0 subject to the convexity constraint. Convex regression is easily extended to concave regression since a concave function is the negative of a convex function.

Convex regression problems occur in a variety of settings. Economic theory often dictates that demand (Varian, 1982), production (Varian, 1984; Allon et al., 2007) and consumer preference (Boyd and Vandenberghe, 2004) functions are concave. In financial engineering, stock option prices often have convexity restrictions (Aït-Sahalia and Duarte, 2003). Stochastic optimization

problems in operations research and reinforcement learning can be solved with response surfaces (Lim, 2010) or value-to-go functions. These exhibit concavity in many settings, like resource allocation (Topaloglu and Powell, 2003; Powell, 2007; Toriello et al., 2010) or stochastic control (Keshavarz et al., 2011). Similarly, efficient frontier methods like data envelopment analysis (Kuosmanen and Johnson, 2010) include convexity constraints. In density estimation, shape restrictions like log-concavity provide flexible estimators without tunable parameters (Cule et al., 2010; Cule and Samworth, 2010; Schuhmacher and Dümbgen, 2010). Finally, in optimization, convex approximations to polynomial constraints are valuable for geometric programming (Kim et al., 2004; Boyd et al., 2007; Magnani and Boyd, 2009).

Although convex regression has been well explored in the univariate setting, the literature remains underdeveloped in the multivariate setting. Methods where an objective function is constrained to the set of convex functions through supporting hyperplane constraints for each pair of observations (Hildreth, 1954; Holloway, 1979; Kuosmanen, 2008; Seijo and Sen, 2011; Lim and Glynn, 2012; Allon et al., 2007) or semidefinite constraints over all observations (Roy et al., 2007; Aguilera and Morin, 2008, 2009; Henderson and Parmeter, 2009; Wang and Ni, 2012) are too computationally demanding for more than a few thousand observations.

In more recent approaches, different methods have been developed. Fitting a convex hull to a smoothed version of the data (Aguilera et al., 2011) scales to larger data sets, but is inefficient for more than 4 or 5 dimensions. Refitting a series of hyperplanes can be done in a frequentist (Magnani and Boyd, 2009) or Bayesian (Hannah and Dunson, 2011) manner. While the Bayesian method does not scale to more than a few thousand observations, the frequentist method scales to much larger data sets but can exhibit unstable behavior. Recent literature is more fully reviewed in Section 2.

In this paper, we introduce the first computationally efficient and theoretically sound multivariate convex regression method: convex adaptive partitioning (CAP). It fits a series of hyperplanes to the data through adaptive partitioning. It relies on an alternate, first-order definition of convexity,

$$f_0(\mathbf{x}_1) \geq f_0(\mathbf{x}_2) + g_0(\mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2), \tag{1}$$

for every $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$, where $g_0(\mathbf{x}) \in \partial f_0(\mathbf{x})$ is a subgradient of f_0 at $\mathbf{x}$. Equation (1) states that a convex function lies above all of its supporting hyperplanes, or subgradients tangent to f_0. Moreover, with enough supporting hyperplanes, f_0 can be approximately reconstructed by taking the maximum over those hyperplanes.

The CAP estimator is formed by adaptively partitioning a set of observations in a method similar to trees with linear leaves (Chaudhuri et al., 1994). Within each subset of the partition, we fit a linear model to approximate the subgradient of f_0 within that subset. Given a partition with K subsets and linear models, $(\alpha_k, \beta_k)_{k=1}^K$, a continuous, convex (concave) function is then generated by taking the maximum (minimum) over the hyperplanes by

$$f_n(\mathbf{x}) = \max_{k \in \{1,\ldots,K\}} \alpha_k + \beta_k^T \mathbf{x}.$$

The partition is refined by a twofold strategy. First, one of the subsets is split along a cardinal direction (say, x_1 or x_3) to grow K. Then, the hyperplanes themselves are used to refit the subsets. A piecewise linear function like f_n induces a partition; a subset is defined as the region where a particular hyperplane is dominant. The refitting step places the hyperplanes in closer alignment with the observations that generated them. This procedure is repeated until all subsets have a minimal number of observations. The CAP estimator is then created by selecting the value of K that balances

fit with complexity using a generalized cross validation method (Golub et al., 1979; Friedman, 1991). We show that CAP is consistent with respect to the ℓ_∞ metric. Because of the dramatic reduction in runtime, CAP opens a new class of problems for study, namely moderate to large problems with convexity or concavity constraints.

2. Literature Review

The literature for convex regression is scattered throughout a variety of fields, including statistics, operations research, economics numerical analysis and electrical engineering. Most methods are designed for the univariate setting, which is closely related to isotonic regression. Univariate methods rely on the ordering implicit to the real line. Setting $x_{i-1} < x_i < x_{i+1}$ for $i = 2, \ldots, n-1$,

$$\frac{f_0(x_i) - f_0(x_{i-1})}{x_i - x_{i-1}} \leq \frac{f_0(x_{i+1}) - f_0(x_i)}{x_{i+1} - x_i}, \qquad i = 2, \ldots, n-1, \tag{2}$$

is equivalent to Equation (1). When f_0 is differentiable, Equation (2) is equivalent to an increasing derivative function.

The oldest and simplest solution method is the least squares estimator (LSE), which produces a piecewise linear estimator by solving a quadratic program with a least squares objective function subject to the constraints in Equation (2) (Hildreth, 1954; Dent, 1973). Although the LSE is completely free of tunable parameters, the estimator is not smooth and can overfit in boundary regions. Consistency, rate of convergence, and asymptotic distribution were shown by Hanson and Pledger (1976), Mammen (1991) and Groeneboom et al. (2001), respectively. Algorithmic methods for solving the quadratic program were given in Wu (1982); Dykstra (1983) and Fraser and Massam (1989).

Splines use linear combinations of basis functions to produce a smooth estimator; in univariate convex regression, an increasing function can be fit to the derivative of the original function. Meyer (2008) and Meyer et al. (2011) used convex-restricted splines with positive parameters in frequentist and Bayesian settings, respectively. Turlach (2005) and Shively et al. (2011) used unrestricted splines with restricted parameters in frequentist and Bayesian settings, respectively. In other methods, Birke and Dette (2007) used convexity constrained kernel regression. Chang et al. (2007) used a random Bernstein polynomial prior with constrained parameters. Due to the constraint on the derivative of f_0, univariate convex regression is quite similar to univariate isotonic regression; see Brunk (1955), Hall and Huang (2001), Neelon and Dunson (2004) and Shively et al. (2009) for examples.

In the multivariate setting Equation (1) cannot be reduced to a set of $n-1$ linear inequalities. Instead, it needs to hold for every pair of points. The multivariate least squares estimator Hildreth (1954); Holloway (1979) solves the quadratic program,

$$\begin{aligned} &\min \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &\text{subject to } \hat{y}_j \geq \hat{y}_i + \mathbf{g}_i^T (\mathbf{x}_j - \mathbf{x}_i), \quad i, j = 1, \ldots, n. \end{aligned} \tag{3}$$

Here, $\hat{y}_i$ and $\mathbf{g}_i$ are the estimated values of $f_0(\mathbf{x}_i)$ and the subgradient of f_0 at $\mathbf{x}_i$, respectively. The estimator f_n^{LSE} is piecewise linear,

$$f_n^{LSE}(\mathbf{x}) = \max_{i \in \{1,\ldots,n\}} \hat{y}_i + \mathbf{g}_i^T (\mathbf{x} - \mathbf{x}_i).$$

The characterization (Kuosmanen, 2008) and consistency (Seijo and Sen, 2011; Lim and Glynn, 2012) of the least squares problem have only recently been studied. The LSE quickly becomes impractical due to its size: Equation (3) has $n(n-1)$ constraints. This results in a computational complexity of $O((p+1)^4 n^5)$ (Monteiro and Adler, 1989), which becomes impractical after one to two thousand observations. It can also severely overfit in boundary regions. In similar approach, Allon et al. (2007) proposed a method based on reformulating the maximum likelihood problem as one minimizing entropic distance, again subject to n^2 linear constraints generated by the dual problem.

An alternative to first order constraints in Equation (1) is second order, or Hessian, constraints. Roy et al. (2007) and Aguilera and Morin (2008, 2009) solved a math program with a least squares objective function and semidefinite constraints through semidefinite programming. Henderson and Parmeter (2009) used kernel smoothing with a restricted Hessian and found a solution with sequential quadratic programming. While these methods are consistent in some cases (Aguilera and Morin, 2008, 2009), they are computationally infeasible for more than about a thousand observations.

Recently, multivariate convex regression methods have been proposed with different approaches. Aguilera et al. (2011) proposed a two step smoothing and fitting process. First, the data were smoothed and functional estimates were generated over an ε-net over the domain. Then the convex hull of the smoothed estimate was used as a convex estimator. Again, although this method is consistent, it is sensitive to the choice of smoothing parameter and does not scale to more than a few dimensions. Hannah and Dunson (2011) proposed a Bayesian model that placed a prior over the set of all piecewise linear models. They were able to show adaptive rates of convergence, but the inference algorithm did not scale to more than a few thousand observations. Koushanfar et al. (2010) transformed the ordering problem associated with shape constrained inference into a combinatorial optimization problem which was solved with dynamic programming; this scales to a few hundred observations.

The work that is closest to CAP is an iterative fitting scheme of Magnani and Boyd (2009). In this method, the data were divided into K random subsets and a linear model was fit within each subset; a convex function was generated by taking the maximum over these hyperplanes. This new function induced a partition over the covariate space, which generated a new collection of K subsets. Again, linear models were fitted and another convex function was produced by taking the maximum over the new hyperplanes. This sequence was repeated until convergence. Although this method usually produces a high quality estimate, it does not always converge and can be unstable.

3. Convex Adaptive Partitioning

A natural way to model a convex function f_0 is through the maximum of a set of K hyperplanes. We do this by partitioning the covariate space and approximating the gradients within each region by hyperplanes generated by the least squares estimator. The covariate space partition and K are chosen through adaptive partitioning. Given a partition $\{A_1,\ldots,A_K\}$ of $\mathcal{X}$, an estimate of the gradient for each subset can be created by taking the least squares linear estimate based on all of the observations within that region,

$$(\alpha_k,\beta_k) = \arg\min_{\alpha,\beta} \sum_{i:\mathbf{x}_i\in A_k} \left(y_i - \alpha - \beta^T \mathbf{x}_i\right)^2.$$

A convex function $\hat{f}$ can be created by taking the maximum over $(\alpha_k, \beta_k)_{k=1}^K$,

$$\hat{f}_n(\mathbf{x}) = \max_{k \in \{1,\ldots,K\}} \alpha_k + \beta_k^T \mathbf{x}.$$

Adaptive partitioning models with linear leaves have been proposed before; see Chaudhuri et al. (1994), Chaudhuri et al. (1995), Alexander and Grimshaw (1996), Nobel (1996), Dobra and Gehrke (2002), Györfi et al. (2002) and Potts and Sammut (2005) for examples. In most of these cases, the partition is created by adaptively refining an existing partition by dyadic splitting of one subset along one dimension. That is, all data is initially placed within a single subset, which is then split into two new subsets along a single dimension, for example at $x_1 = 5$. The split dimension and value is chosen in a way that minimizes local error within the subset, through impurity (Chaudhuri et al., 1994) or mean squared error minimization (Alexander and Grimshaw, 1996). The SUPPORT algorithm of Chaudhuri et al. (1994) computes test statistics for the difference between the means and variances of the residuals and selects the split with the smallest associated $p-$value. Splitting is continued within a subset until a terminal level of purity or a minimal number of observations is reached in that subset; however, SUPPORT uses a cross-validation based method as a stopping rule. Once a full tree has been created, it is pruned using a variety of cross-validation based methods that aim to remove individual leaves or branches to produce the most simple tree that represents the data well; see Breiman et al. (1984) and Quinlan (1993) for pruning methods.

There are two problems that arise when a piecewise linear additive function,

$$f^*(\mathbf{x}) = \sum_{k=1}^{K} \left(\alpha_k + \beta_k^T \mathbf{x}\right) \mathbf{1}_{\{\mathbf{x} \in A_k\}},$$

is changed into a piecewise linear maximization function, like $\hat{f}$. First, a split that minimizes local error does not necessarily minimize global error for $\hat{f}$. This is easily remedied by selecting splits based on minimizing global error. The second problem is more difficult: the linear models often act in areas over which they were not estimated.

The piecewise linear max function, f_n, generates a new partition, $\{A'_1, \ldots, A'_K\}$, by

$$A'_k = \left\{\mathbf{x} \in \mathcal{X} : \alpha_k + \beta_k^T \mathbf{x} > \alpha_j + \beta_j^T \mathbf{x}, \forall\ j \neq k\right\}.$$

The partition $\{A_1, \ldots, A_K\}$ is not necessarily the same as $\{A'_1, \ldots, A'_K\}$. We can use this new partition to refit the hyperplanes and produce a significantly better estimate. A graphical representation is given in Figure 1.

Refitting hyperplanes in this manner can be viewed as a Gauss-Newton method for the non-linear least squares problem (Magnani and Boyd, 2009),

$$\text{minimize} \sum_{i=1}^{n} \left(y_i - \max_{k \in \{1,\ldots,K\}} \left(\alpha_k + \beta_k^T \mathbf{x}_i\right)\right)^2.$$

Similar methods for refitting hyperplanes have been proposed in Breiman (1993) and Magnani and Boyd (2009). However, repeated refitting may not converge to a stationary partition and is sensitive to the initial partition.

Convex adaptive partitioning uses adaptive partitioning with linear leaves to fit a convex function that is defined as the maximum over the set of leaves. The adaptive partitioning itself differs from

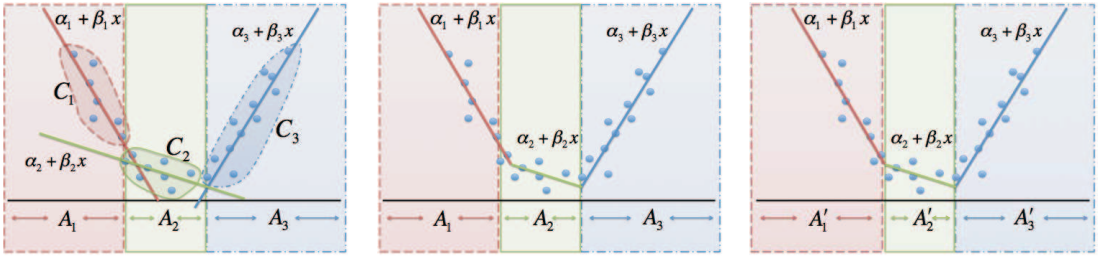


Figure 1: The original space partition A and accompanying data partition C with hyperplanes fit according to that partition (left), the convex estimator based on those hyperplanes; some points are not represented by the hyperplane they were used to fit (center), and subsets refit based on the hyperplanes (right).

previous methods in order to fit piecewise linear maximization functions. Partitions are refined in two steps. First, candidate splits are generated through dyadic splits of existing partitions. These are evaluated and the one that minimizes global error is greedily selected. Second, the new partition is then refit. Although simple, these rules, and refitting in particular, produce large gains over naive adaptive partitioning methods; empirical results are discussed in Section 6.

Most other adaptive partitioning methods use backfitting or pruning to select the tree or partition size. Due to the construction of the CAP estimator, we cannot locally prune and so instead we rely on model selection criteria. We derive a generalized cross-validation method for this setting that is used to select K. This is discussed in Section 5.

3.1 The Algorithm

We now introduce some notation required for convex adaptive partitioning. When presented with data, a partition can be defined over the covariate space (denoted by $\{A_1,\ldots,A_K\}$, with $A_k \subseteq \mathcal{X}$) or over the observation space (denoted by $\{C_1,\ldots,C_K\}$, with $C_k \subseteq \{1,\ldots,n\}$). The observation partition is defined from the covariate partition,

$$C_k = \{i : \mathbf{x}_i \in A_k\}, \quad k = 1,\ldots,K.$$

The relationship between these is shown in Figure 1. CAP proposes and searches over a set of models, $M_1,\ldots,M_K$. A model M_k is defined by: 1) the covariate partition $\{A_1,\ldots,A_K\}$, 2) the corresponding observation partition, $\{C_1,\ldots,C_K\}$, and 3) the hyperplanes $(\alpha_j,\beta_j)_{j=1}^K$ fit to those partitions.

The CAP algorithm progressively refines the partition until each subset cannot be split without one subset having fewer than a minimal number of observations, n_{min}. This value is chosen to balance increasing model complexity against accurate local model fit and computational complexity. When a relatively small number of observations is used to fit local linear models, the local models tend to fit noise. This is particularly problematic with linear models, which can predict extreme values based on overfit models. The issue is aggravated when the estimator is defined as a max over local linear models, which can be dominated by a few extreme values; it can cause instability in the estimator of Magnani and Boyd (2009). Therefore, we choose a conservative value for n_{min}, which

admits logarithmic partition growth,

$$n_{min} = \min\left\{\frac{n}{D\log(n)}, 2(d+1)\right\}.$$

Here D is a log scaling factor, which acts to change the base of the log operator. We briefly outline the CAP algorithm below.

3.1.1 CONVEX ADAPTIVE PARTITIONING (CAP)

1. **Initialize.** Set $K = 1$; place all observations into a single observation subset, $C_1 = \{1, \ldots, n\}$; $A_1 = \mathcal{X}$; this defines model M_1.

2. **Split.** Refine partition by splitting a subset.

   a. *Generate candidate splits.* Generate candidate model $\hat{M}_{kj\ell}$ by 1) fixing a subset k, 2) fixing a dimension j, 3) dyadically dividing the data in subset k and dimensions j according to knot a_ℓ. This is done for L knots, all p dimensions and K subsets.

   b. *Select split.* Choose the model M_{K+1} from the candidates that minimizes global mean squared error on the training set and satisfies $\min_k |C_k| \geq n_{min}$. Set $K = K + 1$.

3. **Refit.** Use the partition induced by the hyperplanes to generate model M'_K. Set $M_K = M'_K$ if for every subset C'_k in M'_K, $|C'_k| \geq n_{min}$.

4. **Stopping conditions.** If for every subset C_k in M_k, $|C_k| < 2n_{min}$, stop fitting and proceed to step 5. Otherwise, go to step 2.

5. **Select model size.** Each model M_k creates an estimator,

$$f_k(\mathbf{x}) = \max_{j \in \{1,\ldots,k\}} \alpha_j + \beta_j^T \mathbf{x}.$$

   Use generalized cross-validation on the estimators to select final model M^* from $\{M_k\}_{k=1}^K$.

3.2 Splitting Rules

To split, we create a collection of candidate models by splitting a single subset into two subsets. We create models for every subset and search along every cardinal direction by splitting the data along that direction. For a fixed dimension j and subset k, let x_{min}^{jk} be the minimum value and x_{max}^{jk} be the maximum value of the covariates in this subset and dimension. Let $0 < a_1 < \cdots < a_L < 1$ be a set of evenly spaced knots that represent the proportion between x_{min}^{jk} and x_{max}^{jk}.

We create model $\hat{M}_{jk\ell}$ by 1) fixing subset $k \in \{1, \ldots, K\}$, and 2) fixing dimension $j \in \{1, \ldots, p\}$.

$$x_{min}^{jk} = \min\{x_{ij} : i \in C_k\}, \qquad x_{max}^{jk} = \max\{x_{ij} : i \in C_k\}.$$

Use the weighted average $b_{jk\ell} = a_\ell x_{min}^{jk} + (1 - a_\ell) x_{max}^{jk}$ to split C_k and A_k in dimension j. Set

$$C'_k = \{i : i \in C_k, x_{ij} \leq b_{jk\ell}\}, \qquad C'_{K+1} = \{i : i \in C_k, x_{ij} > b_{jk\ell}\},$$
$$A'_k = \{\mathbf{x} : \mathbf{x} \in A_k, x_j \leq b_{jk\ell}\}, \qquad A'_{K+1} = \{\mathbf{x} : \mathbf{x} \in A_k, x_j > b_{jk\ell}\}.$$

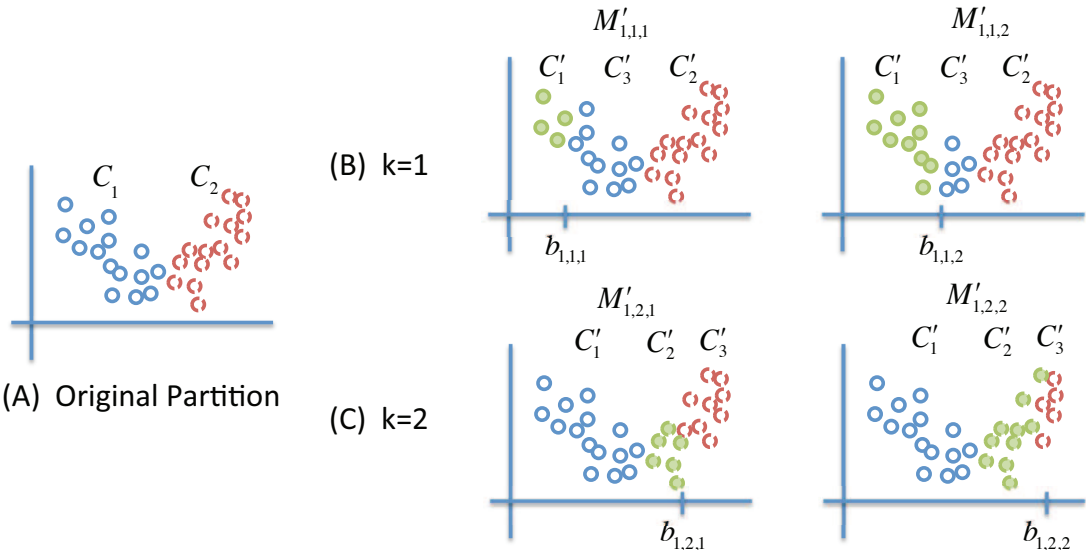


Figure 2: (A) The original observation partition C for M_2, (B) new splits generated from the subset C_1, and (C) new splits generated from the subset C_2. Since there is only one dimension, we fix $j = 1$.

These define new subset and covariate partitions, $C'_{1:K+1}$ and $A'_{1:K+1}$ where $C'_{k'} = C_{k'}$ and $C'_{k'} = C_{k'}$ for $k' \neq k$. See Figure 2 for an example. Fit hyperplanes $(\hat{\alpha}_k, \hat{\beta}_k)_{k=1}^{K+1}$ in each of the subsets. The triplet of observation partition $C'_{1:K+1}$, covariate partition, $A'_{1:K+1}$, and set of hyperplanes $(\hat{\alpha}_k, \hat{\beta}_k)_{k=1}^{K+1}$ defines the model $M'_{jk\ell}$. This is done for $k = 1,\ldots,K$, $j = 1,\ldots,p$ and $\ell = 1,\ldots,L$. After all models are generated, set $K = K+1$.

We note that any models where $\min_k |C'_k| < n_{min}$ are discarded. If all models are discarded in one subset/dimension pair, we produce a model by splitting on the subset median in that dimension.

3.3 Split Selection

We select the model $M'_{jk\ell}$ that gives the smallest *global* error. Let $(\alpha_i^{jk\ell}, \beta_i^{jk\ell})_{i=1}^{K}$ be the hyperplanes associated with $M'_{jk\ell}$ and let

$$\hat{f}^{jk\ell}(\mathbf{x}) = \max_{i\in\{1,\ldots,K\}} \alpha_i^{jk\ell} + \beta_i^{jk\ell T}\mathbf{x}$$

be its estimator. We set the model M_K to be the one that minimizes global mean squared error,

$$M_K = \left\{ \hat{M}_{jk\ell} : (j,k,\ell) = \arg\min_{j,k,\ell} \frac{1}{n}\sum_{i=1}^{n} \left(y_i - \hat{f}^{jk\ell}(\mathbf{x}_i)\right)^2 \right\}.$$

Set $\hat{f}_K$ to be the minimal estimator. We note that M_K may not be unique, however this seldom occurs in practice.

3.4 Refitting

We refit by using the partition induced by the hyperplanes. Let $(\alpha_{1:K}, \beta_{1:K})$ be the hyperplanes associated with M_K. Refit the partitions by

$$C'_k = \{\mathbf{x}_i : \alpha_k + \beta_k^T \mathbf{x}_i \geq \alpha_j + \beta_j^T \mathbf{x}_i, j \neq k\}$$

for $k = 1, \ldots, K$. The covariate partition, $A'_{1:K}$ is defined in a similar manner. Fit hyperplanes in each of those subsets. Let M'_K be the model generated by the partition $C'_1, \ldots, C'_K$. Set $M_K = M'_K$ if $|C'_k| \geq n_{min}$ for all k.

3.5 Stopping Criteria

Stopping criteria are similar to those in tree-based models (Nobel, 1996; Györfi et al., 2002). That is, the model stops when there are not enough observations within each subset of leaf to generate any further candidate splits,

$$|C_k| \leq 2n_{min}$$

for $k = 1, \ldots, K$. After fitting to termination the final model size, however, is chosen through a pruning method discussed Section 5.

3.6 Tunable Parameters

CAP has two tunable parameters, L and n_{min}. L specifies the number of knots used when generating candidate models for a split. Its value is tied to the smoothness of f_0 and after a certain value, usually 5 to 10 for most functions, higher values of L offer little fitting gain.

We choose a minimal subset size, n_{min}, that admits at most $O(\log(n))$subsets. A parameter D is used to specify a minimum subset size, $n_{min} = n/(D\log(n))$. Here D transforms the base of the logarithm from e into $\exp(1/D)$. We have found that $D = 3$ (implying base ≈ 1.4) is a good choice for most problems.

Increases in either of these parameters increase the computational time. Sensitivity to these parameters, both in terms of predictive error and computational time, is empirically examined in Appendix B.

3.7 Computational Efficiency

Each round of CAP requires $O(dKL)$ regressions to be fit for model proposal. Since observations are moved from one side of a threshold to another within each leaf, an efficient method is to maintain and update parameters and the sum of squares and cross products within each leaf. Alternately, a *QR* decomposition may be maintained and updated for each leaf (Alexander and Grimshaw, 1996). Unlike treed linear models, all linear models need to be refit for each round of CAP.

4. Consistency

Consistency for CAP can be shown in a related manner to consistency for other adaptive partitioning models, like CART (Breiman et al., 1984), treed linear models (Chaudhuri et al., 1994) and other variants (Nobel, 1996; Györfi et al., 2002). We take a two-step approach, first showing consistency for the mean function and first derivatives of a more traditional treed linear model based on CAP under the ℓ_∞ metric and then we use that to show consistency for the CAP estimator itself.

Letting M_n^* be the model for the CAP estimate after n observations, define the discontinuous piecewise linear estimate based on M_n^*,

$$f_n^*(\mathbf{x}) = \sum_{k=1}^{K_n} \left(\alpha_k + \beta_k^T \mathbf{x}\right) \mathbf{1}_{\{\mathbf{x} \in A_k\}},$$

where K_n is the partition size, $A_1, \ldots, A_{K_n}$ are the covariate partitions and $(\alpha_k, \beta_k)_{k=1}^{K_n}$ are the hyperplanes associated with M_n^*. Let $f_n(\mathbf{x})$ be the CAP estimator based on M_n^*,

$$f_n(\mathbf{x}) = \max_{k \in \{1,\ldots,K_n\}} \alpha_k + \beta_k^T \mathbf{x}.$$

Each subset A_k has an associated diameter, $d_{nk} = \sup_{\mathbf{x}_1, \mathbf{x}_2 \in A_k} ||x_1 - x_2||_2$. Define the empirical covariate mean for subset k as $\bar{x}_k = \frac{1}{|C_k|} \sum_{i \in C_k} \mathbf{x}_i$. For $\mathbf{x}_i \in A_k$, define

$$\Gamma_i = \begin{bmatrix} [1, \ldots, 1] \\ d_{nk}^{-1} (\mathbf{x}_i - \bar{\mathbf{x}}_k) \end{bmatrix}, \qquad G_k = \sum_{i \in C_k} \Gamma_i \Gamma_i^T.$$

Note that $(\alpha_k, \beta_k) = G_k^{-1} \sum_{i \in C_k} \Gamma_i y_i$ whenever G_k is nonsingular.

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be i.i.d. random variables. We make the following assumptions:

A1. $\mathcal{X}$ is compact and f_0 is Lipschitz continuous and continuously differentiable on $\mathcal{X}$ with Lipschitz parameter ζ.

A2. There is an $a > 0$ such that $\mathbb{E}\left[e^{a|Y - f_0(x)|} \,|\, \mathbf{X} = \mathbf{x}\right]$ is bounded on $\mathcal{X}$.

A3. Let λ_k be the smallest eigenvalue of $|C_k|^{-1} G_k$ and $\lambda_n = \min_k \lambda_k$. Then λ_n remains bounded away from 0 in probability as $n \to \infty$.

A4. The diameter of the partition $\max_k d_{nk}^{-1} \to 0$ in probability as $n \to \infty$.

A5. The number of observations in each subset satisfies $\min_{k=1,\ldots,K_n} |C_k| > d_{nk}^{-1} \sqrt{n \log(n)}$ in probability as $n \to \infty$.

Assumptions **A1.** and **A2.** place regularity conditions on f_0 and the noise distribution, respectively. Assumption **A3.** is a regularity condition on the covariate distribution to ensure the uniqueness of the linear estimates. Assumption **A4.** is a condition that can be included in the algorithm and checked along with the subset cardinality, $|C_k|$. If $\mathcal{X}$ is given, it can be computed directly, otherwise it can be approximated using $\{\mathbf{x}_i : i \in C_k\}$. Assumption **A5.** ensures that there are enough observations in the terminal nodes to fit the linear models.

To show consistency of f_n under the ℓ_∞ metric, we first show consistency of f_n^* and its derivatives under the ℓ_∞ metric in Theorem 1. This is similar to Theorem 1 of Chaudhuri et al. (1994) for treed linear models, although we need to modify it to allow partitions with an arbitrarily large number of faces.

Theorem 1 *Suppose that assumptions* **A1.** *through* **A5.** *hold. Then,*

$$\max_{k=1,\ldots,K_n} \sup_{\mathbf{x} \in A_k} \left|\alpha_k + \beta_k^T \mathbf{x} - f_0(\mathbf{x})\right| \to 0, \qquad \max_{k=1,\ldots,K_n} \sup_{\mathbf{x} \in A_k} ||\beta_k - \nabla f_0(\mathbf{x})||_\infty \to 0$$

in probability as $n \to \infty$.

The CAP algorithm is similar to the SUPPORT algorithm of Chaudhuri et al. (1994), except the refitting step of CAP allows partition subsets to be polyhedra with up to K_n faces. Theorem 1 is analogous to Theorem 1 of Chaudhuri et al. (1994); to prove our theorem, we modify parts of the proof in Chaudhuri et al. (1994) that rely on a fixed number of polyhedral faces. The proof is given in Appendix A.

Using the results from Theorem 1, extension to consistency for f_n under the ℓ_∞ metric is fairly simple; this is given in Theorem 2.

Theorem 2 *Suppose that assumptions* **A1.** *through* **A5.** *hold. Then,*

$$\sup_{\mathbf{x}\in\mathcal{X}} |f_n(\mathbf{x}) - f_0(\mathbf{x})| \to 0$$

in probability as $n \to \infty$.

The proof follows immediately from Theorem 1 and some algebra. Details are given in the Appendix A.

5. Generalized Cross-Validation

The terminal model produced by CAP can overfit the data. As a fast approximation to leave-one-out cross-validation, we use generalized cross-validation (GCV) (Golub et al., 1979; Friedman, 1991) to select the best model from all of those produced by CAP, $M_1,\ldots,M_K$. A given model M_K is generated by a collection of K linear models. In linear regression, GCV relies on the following approximation

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f_{-i}(\mathbf{x}_i))^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - f_n(\mathbf{x}_i)}{1-H_{ii}}\right)^2 \approx \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - f_n(\mathbf{x}_i)}{1-\mathrm{Tr}(H)}\right)^2, \tag{4}$$

where H_{ii} is the i^{th} diagonal element of the hat matrix, $\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, $\hat{f}_{-i}$ is the estimator conditioned on all of the data minus element i. We note that $\mathrm{Tr}(H)$ is sometimes approximated by the degrees of freedom divided by the number of observations.

The model M_K is defined by $C_1,\ldots,C_K$, the partition, and the hyperplanes $(\alpha_k,\beta_k)_{k=1}^K$, which were generated by the partition. Let $(\alpha_k^{(-i)},\beta_k^{(-i)})_{k=1}^K$ be the collection of hyperplanes generated when observation i is removed; notice that if $i \in C_k$, only (α_k,β_k) changes. Let $\hat{f}_{-iK}$ be the estimator for model M_K with observation i removed. Using the derivation in Equation (4),

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{f}_{-iK}(\mathbf{x}_i)\right)^2 &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \max_{k\in\{1,\ldots,K\}} \alpha_k^{(-i)} + \beta_k^{(-i)^T}\mathbf{x}_i\right)^2, \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \alpha_{k(i)} - \beta_{k(i)}^T\mathbf{x}_i}{1 - H_{ii}^{k(i)}\mathbf{1}_{\{i\in C_{k(i)}\}}}\right)^2 \\
&\approx \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \alpha_{k(i)} - \beta_{k(i)}^T\mathbf{x}_i}{1 - \mathrm{Tr}(H^{k(i)})\mathbf{1}_{\{i\in C_{k(i)}\}}}\right)^2,
\end{aligned} \tag{5}$$

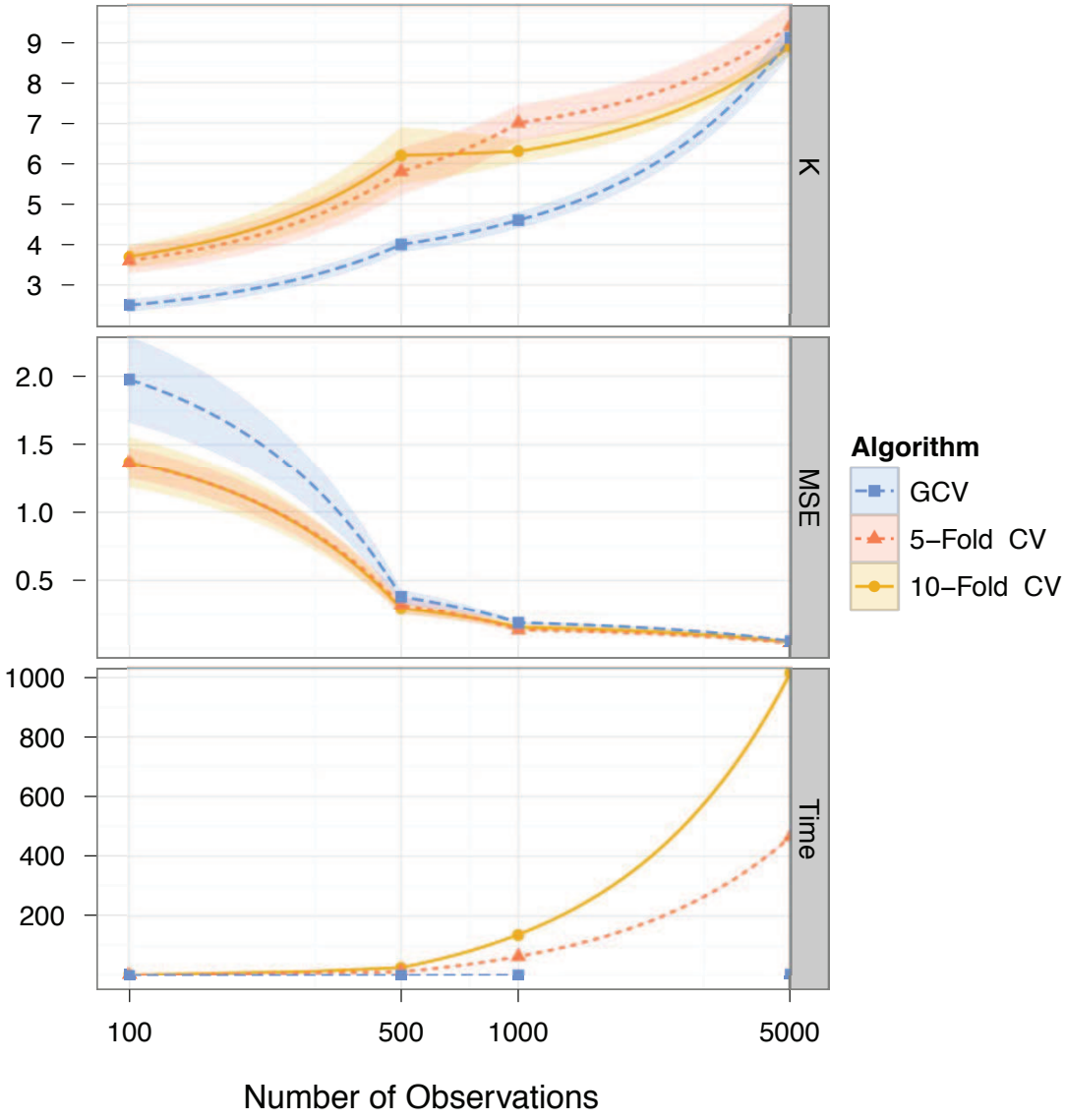


Figure 3: Log-continuous plots for number of observations vs. K (top), MSE (middle), and runtime in seconds (bottom) for GCV, 5-fold and 10-fold cross validation, plus/minus one standard error. Data were generated from 10 i.i.d. training sets with $\mathbf{x} \sim N_5(0,I)$, $y = (x_1 + .5x_2 + x_3)^2 - x_4 + .25x_5^2 + \varepsilon$, and $\varepsilon \sim N(0,1)$.

where, in a slight abuse of notation, H_{ii}^k is the diagonal entry of the hat matrix for subset k corresponding to element i, and

$$k(i) = \arg \max_{k \in \{1,\ldots,K\}} \frac{\alpha_k + \beta_k^T \mathbf{x}_i}{1 - \mathrm{Tr}(H^k)\mathbf{1}_{\{i \in C_k\}}}.$$

To select K, we find the K that minimizes the right hand side of Equation (5). Although more computationally intensive than GCV in linear models, the computational complexity for CAP GCV is similar to that of the CAP split selection step.

We empirically compared GCV selection of K with 5- and 10-fold cross validation selection of K. GCV tends to select a smaller K than full cross validation, particularly on smaller problems. Predictive results, however, are comparable for moderate to large problem sizes ($n \geq 5,000$) while the runtime of GCV is orders of magnitude less than 5- and 10-fold cross validation. We should expect more discrepancy between cross-validation and GCV on smaller problems because GCV relies on an asymptotic approximation. In these cases, full cross validation selection of K may be worthwhile. Representative results are given in Figure 3.

We can use generalized cross-validation to create a more efficient stopping rule for CAP. We note that GCV scores are often unimodal in K. Instead of fully growing the tree, we stop splitting after the score has increased twice in a row. The resulting algorithm is called Fast CAP; details are given in Appendix B.

6. Empirical Analysis

We compare shape constrained and unconstrained regression methods across a set of convex regression problems: two synthetic regression problems, predicting mean weekly wages and value function approximation for pricing basket options.

6.1 Synthetic Regression Problems

We apply CAP to two synthetic regression problems to demonstrate predictive performance and analyze sensitivity to tunable parameters. The first problem has a non-additive structure, high levels of covariate interaction and moderate noise, while the second has a simple univariate structure embedded in a higher dimensional space and low noise. Low noise or noise free problems often occur when a highly complicated convex function needs to be approximated by a simpler one (Magnani and Boyd, 2009).

6.1.1 PROBLEM 1

Here $\mathbf{x} \in \mathbb{R}^5$. Set

$$y = (x_1 + .5x_2 + x_3)^2 - x_4 + .25x_5^2 + \varepsilon,$$

where $\varepsilon \sim N(0,1)$. The covariates are drawn from a 5 dimensional standard Gaussian distribution, $N_5(0,I)$.

6.1.2 PROBLEM 2

Here $\mathbf{x} \in \mathbb{R}^{10}$. Set

$$y = \exp\left(\mathbf{x}^T\mathbf{q}\right) + \varepsilon,$$

where $\mathbf{q}$ was randomly drawn from a Dirichlet(1,...,1) distribution,

$$\mathbf{q} = (0.0680, 0.0160, 0.1707, 0.1513, 0.1790, 0.2097, 0.0548, 0.0337, 0.0377, 0.0791)^T.$$

We set $\varepsilon \sim N(0, 0.1^2)$. The covariates are drawn from a 10 dimensional standard Gaussian distribution, $N_{10}(0,I)$.

6.1.3 PREDICTIVE PERFORMANCE AND RUNTIMES

We compared the performance of CAP and Fast CAP to other regression methods on problems 1 and 2. We implemented the following shape constrained algorithms: the least squares regression (LSE) using `cvx` (Grant and Boyd, 2012, 2008), and the linear refitting algorithm of Magnani and Boyd (2009). The general methods included Gaussian processes (Rasmussen and Williams, 2006) using `gpml` in Matlab, tree regression with constant values in the leaves using `classregtree` in Matlab, multivariate adaptive regression splines (MARS) (Friedman, 1991) using `ARESlab` in Matlab, and support vector machines (SVMs) using the `e1071` package in `R`.

For CAP and Fast CAP, we set the parameters to $D = 3$ and $L = 10$; the sensitivity to these parameters is examined in Appendix C. The parameter K was chosen by GCV for CAP. In Fast CAP, the number of random search directions was set to be $p = \min(d, 10)$. All methods were given a maximum runtime of 90 minutes, after which the results were discarded. Methods were run on 10 random training sets and tested on the same testing set of 10,000 random covariates. Average runtimes and predictive performance are shown in Figure 4.

Non-convex regression methods performed poorly compared to shape restricted methods, particularly in the higher noise setting. Amongst the shape restricted methods, only CAP and Fast CAP had consistently low predictive error. The method of Magnani and Boyd (2009) can become unstable, which is seen in problem 1. Surprisingly, the LSE had high predictive error. This can be attributed to overfitting, particularly in the boundary regions. A demonstration is given in Figure 5. Although CAP and Fast CAP had similar predictive performance, their runtimes often differed by an order of magnitude with the largest differences on the biggest problem sizes. Based on this performance, we would suggest using Fast CAP on larger problems.

We note that the empirical rate of convergence for CAP and Fast CAP is much faster than would be predicted by minimax convergence rates. The results, however, are consistent with rates that adapt to an underlying linear subspace; this is examined in Appendix D.

6.1.4 CAP AND TREED LINEAR MODELS

Treed linear models are a popular method for regression and classification. They can be easily modified to produce a convex regression estimator by taking the maximum over the linear leaves. CAP differs from existing treed linear models in how the partition is refined. First, subset splits are selected based on global reduction of error. Second, the partition is refit after a split is made. To investigate the contributions of each step, we compare to treed linear models generated by: 1) local error reduction as an objective for split selection and no refitting, 2) global error reduction as an objective function for split selection and no refitting, and 3) local error reduction as an objective for split selection along with refitting. All estimators based on treed linear models are generated by taking the maximum over the set of linear models in the leaves. We compared the performance of these methods on problems 1 and 2 over 10 different training sets and a single testing set. Average predictive error is displayed in Figure 6.

Global split selection and refitting are both beneficial, but in different ways. Refitting dramatically reduces predictive error, but can add variance to the estimator in noisy settings. Global split selection modestly reduces predictive error but can reduce variance in noisy settings, like problem 1. The combination of the two produces CAP, which has both low variance and high predictive accuracy.

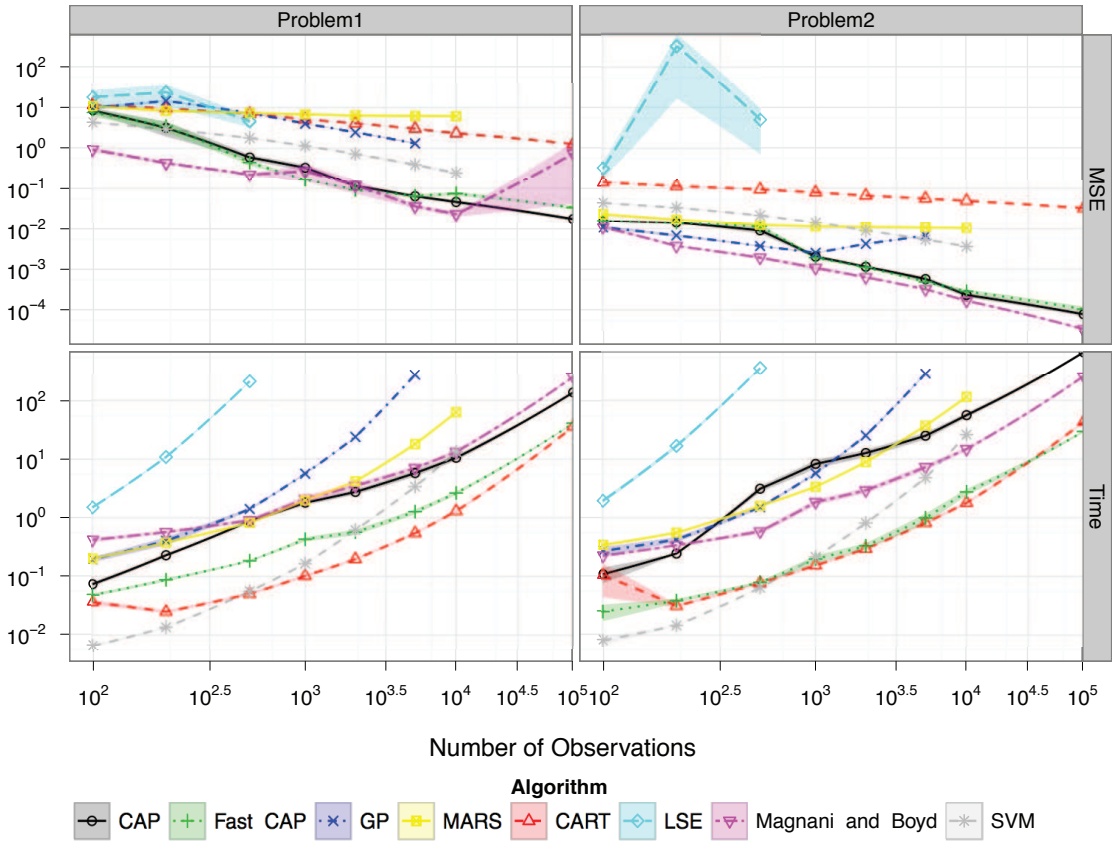


Figure 4: Mean squared error (top) and runtime in seconds (bottom) plus/minus one standard error on problem 1 (left) and problem 2 (right) for CAP, Fast CAP, Gaussian processes, MARS, CART, the least squares estimator, the linear fitting method of Magnani and Boyd (2009) and support vector machines.

6.2 Predicting Weekly Wages

We use shape restricted methods to predict mean weekly wages based on years of education and experience. The data are from the 1988 Current Population Survey (CPS); they originally appeared in Bierens and Ginther (2001) and can be accessed as `ex1029` in the `Sleuth2` package in `R`. The data set contains 25,361 records of weekly wages for full-time, adult, male workers for 1987, along with years experience, years of education, race (either back or white; no others were included in the sample), region, and whether the last job held was part time.

A reasonable assumption for wages is that they are concave in years experience. Each year previously worked should have decreasing returns for average wages until peak earnings are reached, with modest declines afterwards. Indeed, this pattern is seen in Figure 7 when we compare average weekly wages against experience. Wages can also be expected to increase based on education level, but not in a concave or convex fashion. However, concavity can be generated with an exponential transformation of education; this is shown in Figure 7. We therefore used a transformation,

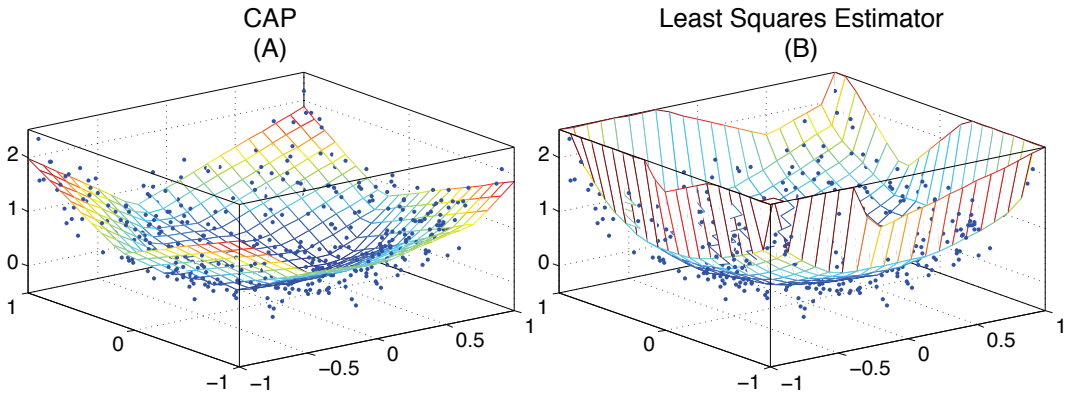


Figure 5: (A) The CAP estimator, and (B) the LSE fit to 500 observations drawn from $y = x_1^2 + x_2^2 + \varepsilon$, where $\varepsilon \sim N(0, 0.25^2)$. The covariates were drawn from a 2 dimensional uniform distribution, Unif$[-1,1]^2$. The LSE was truncated at predicted values of 2.5 for display, although some predicted values reached as high as 4,800 on $[-1,1]^2$.

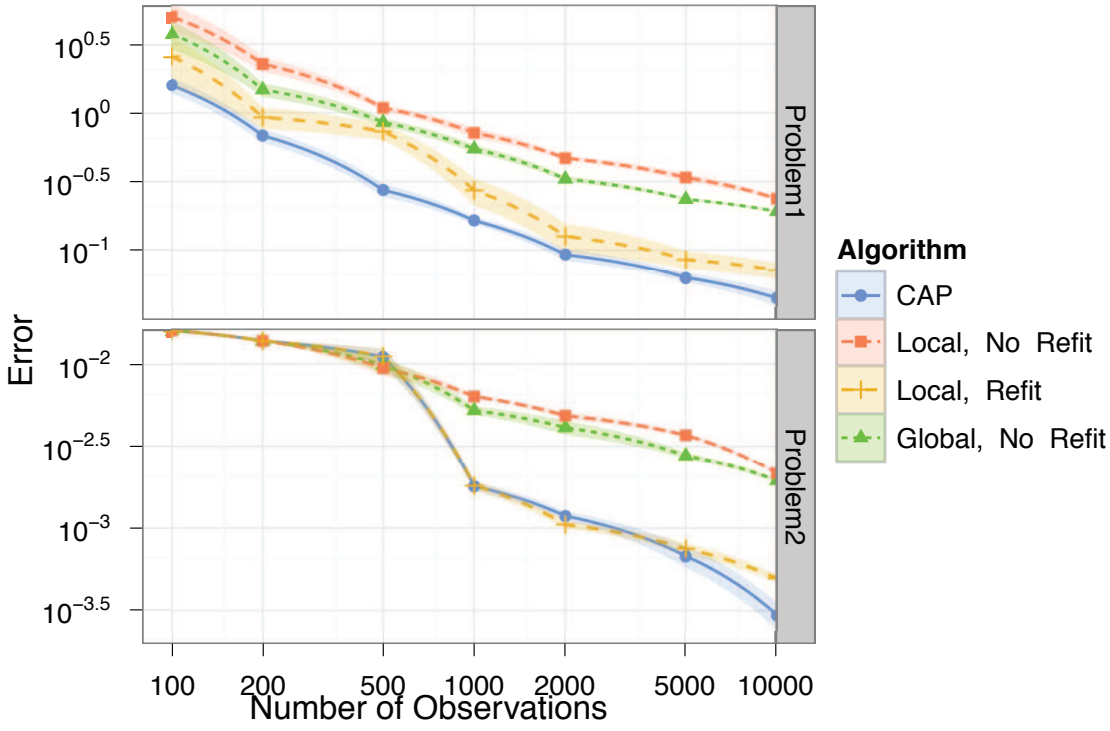


Figure 6: Number of observations (log scale) vs. mean squared error (log scale) plus/minus one standard error for CAP and treed linear models with local split selection, with no refitting; local split selection, with refitting; and global split selection, with no refitting.

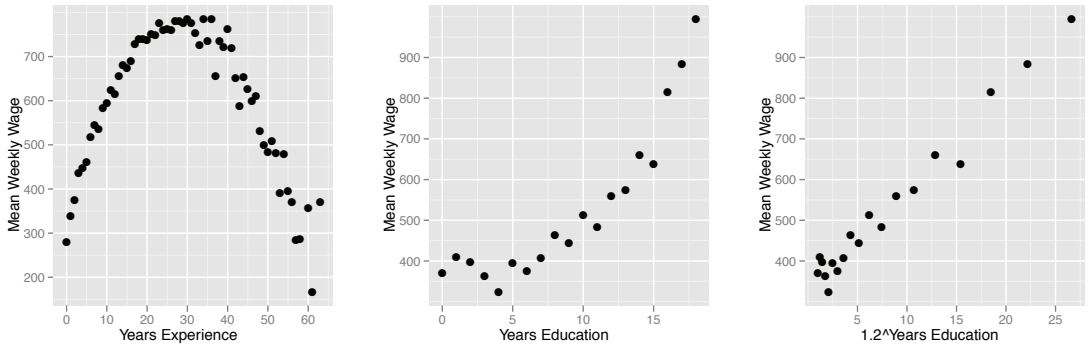


Figure 7: Mean weekly wages vs. years of experience (left), mean weekly wages vs. years of education (center), mean weekly wages vs. $1.2^{\text{years education}}$ (right).

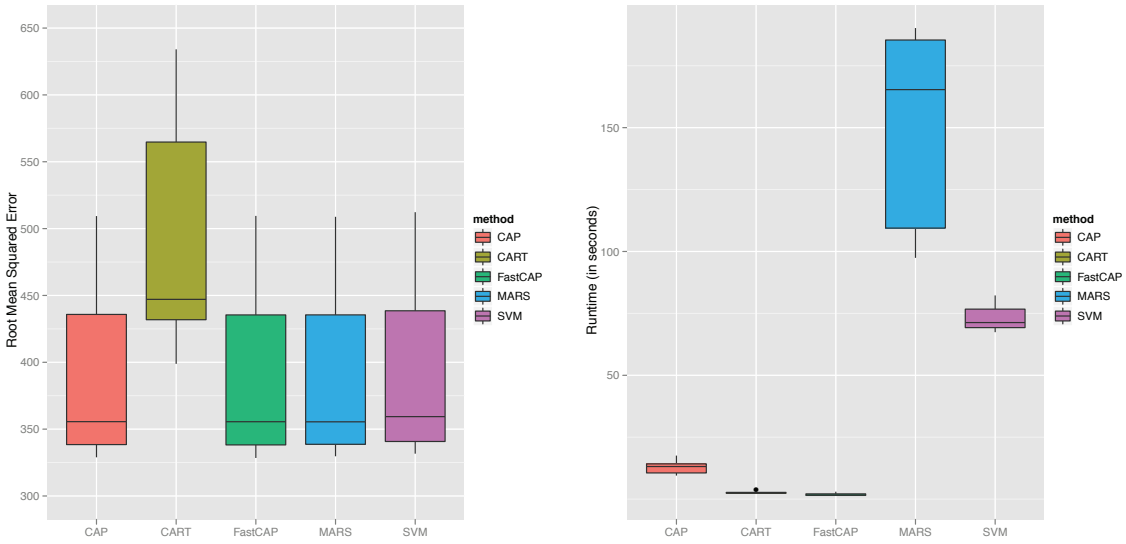


Figure 8: Root mean squared error (RMSE), left, and runtime in seconds, right, for CAP, Fast CAP, CART, MARS, and SVMs for predicting weekly wages based on years experience and years education.

$1.2^{\text{years education}}$, as a covariate. Shape restrictions do not hold with any other covariates, so they are discarded.

We implemented CAP, fast CAP, the linear model of Magnani and Boyd (2009), CART, MARS, and SVMs. Due to the problem size, we did not use Gaussian processes or the least squares estimator. We estimated RMSE through 10-fold cross validation. Results and runtimes are shown in Figure 8 for all methods except Magnani and Boyd (2009). This had a RMSE of 10,156, orders of magnitude larger than other methods, and was hence omitted from the figures.

Method	RMSE	Time
CAP	$\mathbf{385.7 \pm 20.8}$	12.8 ± 0.8
Fast CAP	$\mathbf{385.7 \pm 20.8}$	1.9 ± 0.2
CART	489.6 ± 26.1	2.6 ± 0.2
MARS	385.8 ± 20.7	150.4 ± 12.8
Magnani and Boyd (2009)	10156.0 ± 9765.2	8.0 ± 0.7
SVM	388.9 ± 20.7	72.9 ± 1.6

Table 1: Average RMSE and runtime in seconds, plus/minus one standard error for CAP, Fast CAP, CART, MARS, Magnani and Boyd (2009) and SVMs.

This data set presents difficulties for many methods due to its size ($n > 20,000$) and highly skewed distribution. CAP, Fast CAP, MARS and SVMs all had comparable predictive error rates, while CART produced error rates about 27% higher. The linear fitting method of Magnani and Boyd (2009) occasionally tried to fit outliers with hyperplanes, resulting in about a 2,500% increase in predictive error. This potential instability is one of the largest drawbacks with the method of Magnani and Boyd (2009). In terms of runtimes, Fast CAP and CAP were both significantly faster than any methods that produced comparable results, with runtime reductions of more than 80% over SVMs.

In Figure 9, we compare the predicted functions produced by CAP and SVMs. In areas with small amounts of data, such for people with low education, the SVM produces results that do not match prior information. In the SVM surface, someone with 0 years of experience and 0 years of education is predicted to have about a 150% larger weekly wage than a high school graduate with 0 years of experience and about the same weekly wage as someone with a 4-year college degree and 0 years of experience. By imposing shape constraints, CAP eliminates these types of problems and produces a surface that conforms to prior knowledge.

Unlike the surface produced by SVM regression, the surface produced by CAP is not smooth. A greater degree of smoothness can be added through ensemble methods like bagging (Breiman, 1996) and smearing (Breiman, 2000). Averaging randomized convex estimators produces a new convex estimator; these methods have been explored for approximating objective functions in Hannah and Dunson (2012). A surface produced by smearing CAP is shown on the right in Figure 9. Note that its overall shape is quite similar to the original CAP estimator while most of the sharp edges have been smoothed away.

6.3 Pricing Stock Options

In sequential decision problems, a decision maker takes an action based on a currently observed state of the world based on the current rewards of that action and possible future rewards. Approximate dynamic programming is a modeling method for such problems based on approximating a value-to-go function. Value-to-go functions, or simply "value functions," give the value for each state of the world if all optimal decisions are made subsequently.

Often value functions are known to be convex or concave in the state variable; this is common in options pricing, portfolio optimization and logistics problems. In some situations, such as when a linear program is solved each time period to determine an action, a convex value function *is*

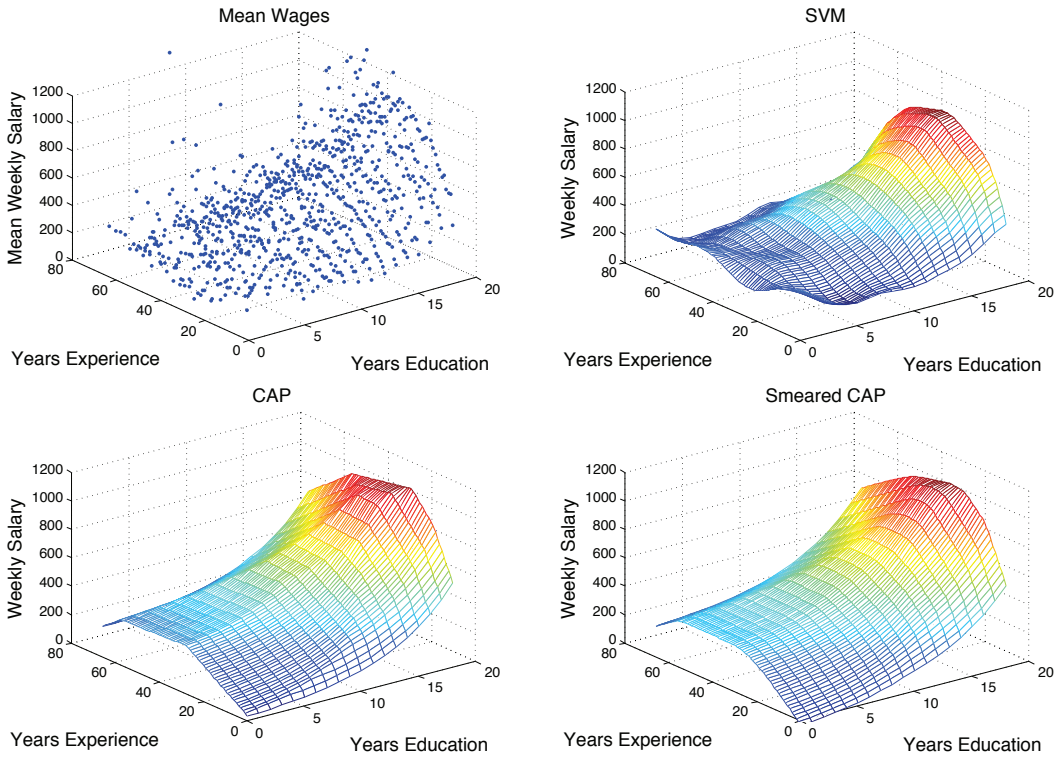


Figure 9: Mean weekly wage based on years of experience and years of education (top left), predicted values using SVM regression (top right), CAP (bottom left), and smeared CAP (bottom right).

required for computational tractability. Convex regression holds great promise for value function approximation in these problems.

To give a simple example for value function approximation, we consider pricing American basket options on the average of M underlying assets. Options give the holder the right—but not the obligation—to buy the underlying asset, in this case the average of M individual assets, for a predetermined strike price R. In an American option, this can be done at any time between the issue date and the maturity date, T. However, American options are notoriously difficult to price, particularly when the underlying asset base is large.

A popular method for pricing American options uses approximate dynamic programming where continuation values are approximated via regression (Carriere, 1996; Tsitsiklis and Van Roy, 1999, 2001; Longstaff and Schwartz, 2001). We summarize these methods as follows; see Glasserman (2004) for a more thorough treatment. The underlying assets are assumed to have the sample path $\{X_1,\ldots,X_T\}$, where $X_t = \{S_1(t),\ldots,S_M(t)\}$ is the set of securities at time t. At each time t, a continuation value function, $\bar{V}_t(X_t)$, is estimated by regressing a value function for the next time period, $\bar{V}_{t+1}(X_{t+1})$, on the current state, X_t. The continuation value is the value of holding the option rather than exercising given the current state of the assets. The value function is defined to

be the max of the current exercise value and the continuation value. Options are exercised when the current exercise value is greater than or equal to the continuation value.

The procedure to estimate the continuation values is as follows (as summarized in Glasserman 2004):

0. Define basket payoff function,

$$h(X_t) = \max\left\{\frac{1}{M}\sum_{k=1}^{M} S_k(t) - R, 0\right\}.$$

1. Sample N independent paths, $\{X_{1j},\ldots,X_{Tj}\}$, $j = 1,\ldots,N$.
2. At time T, set $\bar{V}_T(X_{Tj}) = h(X_{Tj})$.
3. Apply backwards induction: for $t = T-1,\ldots,1$,
   - given $\{\bar{V}_{t+1}(X_{t+1j})\}_{j=1}^N$, regress on $\{X_{tj}\}_{j=1}^N$ to get continuation value estimates $\{\bar{C}_t(X_{tj})\}_{j=1}^N$.
   - set value function,
$$\bar{V}_t(X_{tj}) = \max\left\{h(X_{tj}), \bar{C}_t(X_{tj})\right\}.$$

We use the value function defined by Tsitsiklis and Van Roy (1999).

The regression values are used to create a policy that is implemented on a test set: exercise when the current exercise value is greater than or equal to the estimated continuation value. A good regression model is crucial to creating a good policy.

In previous literature, $\{C_t(X_{tj})\}_{j=1}^N$ has been estimated by regression splines for a single underlying asset (Carriere, 1996), or least squares linear regression on a set of basis functions (Tsitsiklis and Van Roy, 1999; Longstaff and Schwartz, 2001; Glasserman, 2004). Regression on a set of basis functions becomes problematic when X_{tj} is defined over moderate to high dimensional spaces. Well-defined sets of bases such as radial basis functions and polynomials require an exponential number of functions to span the space, while manually selecting basis functions can be quite difficult. Since the expected continuation values are convex in the asset price for basket options, CAP is a simple, nonparametric alternative to these methods.

We compared the following methods: CAP and Fast CAP with $D = 3$, $L = 10$ for both and $P' = \min(M, 10)$, the number of random search directions in Fast CAP; the method of Magnani and Boyd (2009); regression trees with constant leaves using the Matlab function `classregtree`; least squares using the polynomial basis functions

$$(1, S_i(t), S_i^2(t), S_i^3(t), S_i(t)S_j(t), h(X_t)),\ i = 1,\ldots,M,\ j \neq i;$$

ridge regression on the same basis functions with ridge parameter chosen by 10-fold cross-validation each time period from values between 10^{-3} and 10^5.

We compared value function regression methods as follows. We simulated for both $N = 10,000$ and $N = 50,000$ training samples for a 3-month American basket option with a number of underlying assets, M, varying between 1 and 30 using a geometric Brownian motion with a drift of 0.05 and a volatility of 0.10. All assets had correlation 0.5 and starting value 100. The option had strike price 110. Policy values were approximated on 50,000 testing sample paths. An approximate upper bound

was generated using the dual martingale methods of Haugh and Kogan (2004) from value functions generated using polynomial basis functions based on the mean of the assets, $(1, Y_t, Y_t^2, Y_t^3, h(Y_t))$, where $Y_t = 1/M \sum_{i=1}^{M} X_i(t)$, with 2,000 samples. Upper and lower bounds were generated using 5 training and testing sets.

Results are displayed in Figure 10. We found that CAP and Fast CAP gave state of the art performance without the difficulties associated with linear functions, such as choosing basis functions and regularization parameters. We observed a decline in the performance of least squares as the number of assets grew due to overfitting. Ridge regularization greatly improved the least squares performance as the number of assets grew. Tree regression did poorly in all settings, likely due to overfitting in the presence of the non-symmetric error distribution generated by the geometric Brownian motion. These results suggest that CAP is robust even in less than ideal conditions, such as when data have heteroscedastic, non-symmetric error distributions.

Again, we noticed that while the performances of CAP and Fast CAP were comparable, the runtimes were about an order of magnitude different. On the larger problems, runtimes for Fast CAP were similar to those for unregularized least squares. This is likely because the number of covariates in the least squares regression grew like M^2, while all linear regressions in CAP only had M covariates.

7. Conclusions

In this article, we presented convex adaptive partitioning, a computationally efficient, theoretically sound and empirically robust method for regression subject to a convexity constraint. CAP is the first convex regression method to scale to large problems, both in terms of dimensions and number of observations. As such, we believe that it can allow the study of problems that were once thought to be computationally intractable. These include econometrics problems, like estimating consumer preference or production functions in multiple dimensions, approximating complex constraint functions for convex optimization, or creating convex value-to-go functions or response surfaces that can be easily searched in stochastic optimization.

CAP can be extended in a number of ways. First, as demonstrated in Hannah and Dunson (2012), CAP can be used in an ensemble setting—like bagging or smearing—to produce a smoother estimator. Averages of piecewise linear estimators are particularly useful in an optimization setting. They are still piecewise linear and can be searched by a linear program, but have more stable minimum locations than sparse methods like CAP and the linear fitting method of Magnani and Boyd (2009). Second, CAP can be extended to more shape constrained settings, like monotone, concave and semi-convex functions. Monotone, concave functions are concave functions with increasing slopes, which are common in economics. Although CAP is not a general purpose shape constrained inference method, a variant for monotone functions can easily be generated by placing positivity constraints on the parameters of the linear models. Semi-convex functions are convex in some dimensions but unconstrained in others. Variants of CAP could be combined with other nonparametric methods like kernels to produce efficient inference methods for partially convex functions. We believe that the methods supporting the CAP algorithm can bring efficient, theoretically sound inference to a variety of shape constrained problems that are inapproachable with traditional methods.

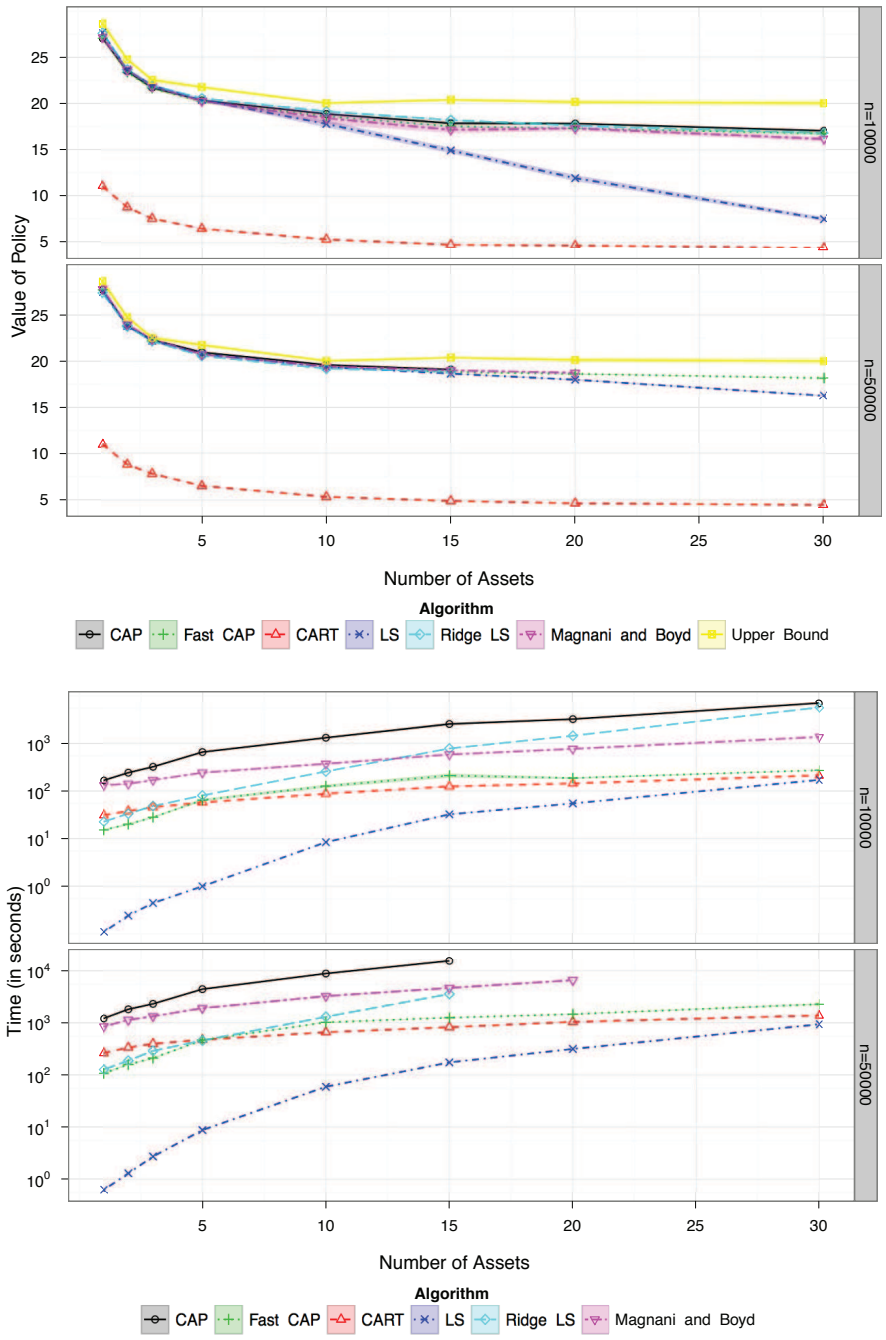


Figure 10: Average values (top) and runtimes (bottom) plus/minus one standard error for pricing America basket options as a function of the number of underlying assets are shown for CAP, Fast CAP, tree regression with constant leaves (CART), the method of Magnani and Boyd (2009), least squares (LS) and ridge regularized least squares value function approximation.

8. Online Supplements

Code for CAP can be downloaded at `http://www.columbia.edu/~lah2178/Research.html` and as an online supplement at the JMLR website.

Acknowledgments

This work was partially supported by grant R01 ES017240 from the National Institute of Environmental Health Sciences (NIEHS) of the National Institutes of Health (NIH). Lauren A. Hannah was partially supported by the Provost's Postdoctoral Fellowship at Duke University.

Appendix A.

The proof of Theorem 1 is essentially identical to the proof of Theorem 1 of Chaudhuri et al. (1994) with a few modifications. The Chaudhuri et al. (1994) results are for an algorithm that splits subsections parallel to axes. By allowing subsets to be determined by the dominating hyperplanes, the subsets are now polyhedral with a maximal number of faces determined by the dimension and maximal number of subsets. To show this, we modify Lemma 12.27 of Breiman et al. (1984).

Proof [Proof of Theorem 1] It is sufficient to show that

$$\max_{k=1,\ldots,K_n} d_{nk}^{-1} \left| [\alpha_k, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k) \right| \to 0$$

in probability, where $\Delta(\bar{\mathbf{x}}_k)$ is the vector of elements $[f_0(\bar{x}_k, d_{nk}^{-1} \frac{\partial}{\partial x_1} f_0(\bar{\mathbf{x}}_k), \ldots, d_{nk}^{-1} \frac{\partial}{\partial x_p} f_0(\bar{\mathbf{x}}_k)]^t$. Let $\alpha_k^0 = \alpha_k - \beta_k^t \bar{\mathbf{x}}_k$. Assumption **A3.** ensures that the matrices D_k for all subsets are nonsingular with probability tending to 1, where $D_k = \sum_{\mathbf{x}_i \in C_k} \Gamma_i \Gamma_i^t$. Letting $Y_i = f_0(\mathbf{x}_i) + \varepsilon_i$, we have

$$[\alpha_k^0, \beta_k]^t = D_k^{-1} \sum_{\mathbf{x}_i \in C_k} \Gamma_i f_0(\mathbf{x}_i) + D_k^{-1} \sum_{\mathbf{x}_i \in C_k} \Gamma_i \varepsilon_i \tag{6}$$

for $k = 1, \ldots, K_n$ with probability tending towards 1. Doing a Taylor expansion of Equation (6), we get

$$[\alpha_k^0, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k) = D_k^{-1} \sum_{i \in C_k} \Gamma_i r(\mathbf{x}_i - \bar{\mathbf{x}}_k) + D_k^{-1} \sum_{i \in C_k} \Gamma_i \varepsilon_i,$$

where $r(\mathbf{x}_i - \bar{\mathbf{x}}_k)$ are the second order and above terms of the Taylor expansion of $f_0(\bar{\mathbf{x}}_k)$. Assumptions **A1.**, **A3.** and **A4.** ensure $\max_{k=1,\ldots,K_n} \left| d_{nk}^{-1} D_k^{-1} \sum_{i \in C_k} \Gamma_i r(\mathbf{x}_i - \bar{\mathbf{x}}_k) \right| \to 0$ in probability as $n \to \infty$. To bound the random error term of Equation (6), we first assume that A_k is fixed. Applying Lemma 12.26 of Breiman et al. (1984) to each component of $d_{nk}^{-1} |C_k|^{-1} \sum_{i \in C_k} \Gamma_i \varepsilon_i$, there exist constants $h_1 > 0$, $h_2 > 0$ and $\gamma_0 > 0$ such that

$$\mathbb{P}\left(d_{nk}^{-1} \left| |C_k|^{-1} \sum_{i \in C_k} \Gamma_i \varepsilon_i \right| > \gamma \right) \leq h_1 e^{-h_2 d_{nk}^2 |C_k| \gamma^2}, \tag{7}$$

whenever $\gamma \leq \gamma_0$. Modifying Lemma 12.27 of Breiman et al. (1984) to account for the greater VC dimension of the subsets, we note **A5.** bounds the number of polyhedral faces for each subset to be

$K_n(p+2)$. Following the proof of 12.27, for m_n such that $m_n/\log(n) \to \infty$, we use Equation (7) to show

$$\begin{aligned}\mathbb{P}\left(\left|[\alpha_k^0, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k)\right| \geq \gamma | \mathbf{x}_{1:n}\right) &\leq h_1 e^{-h_2\gamma^2 n d_{nk}^2 |C_k|/n}, \\ &\leq h_1 e^{-h_2\gamma^2 m_n \log(n)}, \\ &= h_1 n^{-h_2\gamma^2 m_n}\end{aligned}$$

on the event $d_{nk}^2|C_k|/n \geq m_n \log(n)/n$. Using the VC dimension of the partition,

$$\begin{aligned}&\mathbb{P}\left(\left|[\alpha_k^0, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k)\right| \geq \gamma \text{ for each } A_k \text{ and } d_{nk}^2|C_k| \geq m_n D \log(n)\right) \\ &\leq h_1 2^{1+K_n(p+2)} n^{K_n(p+2)-h_2\gamma^2 m_n}.\end{aligned}$$

Since **A5.** ensures that $m_n \to \infty$, the result holds.

■

Proof [Proof of Theorem 2] Fix $\varepsilon > 0$; let $d_{\mathcal{X}}$ be the diameter of $\mathcal{X}$. Choose N such that for every $n \geq N$

$$\mathbb{P}\left\{\max_{k=1,\ldots,K_n} \sup_{\mathbf{x}\in A_k} \left|\alpha_k + \beta_k^T \mathbf{x} - f_0(\mathbf{x})\right| > \frac{\varepsilon}{\zeta d_{\mathcal{X}}}\right\} < \varepsilon/2,$$

$$\mathbb{P}\left\{\max_{k=1,\ldots,K_n} \sup_{\mathbf{x}\in A_k} ||\beta_k - \nabla f_0(\mathbf{x})||_\infty > \frac{\varepsilon}{\zeta d_{\mathcal{X}}}\right\} < \varepsilon/2.$$

Fix a δ net over $\mathcal{X}$ such that at least one point of the net sits in A_k for each $k = 1, \ldots, K$. Let n_δ be the number of points in the net and let $\mathbf{x}_i^\delta$ be a point. Then,

$$\begin{aligned}\mathbb{P}\left\{\sup_{x\in\mathcal{X}} |f_n(\mathbf{x}) - f_0(\mathbf{x})| > \varepsilon\right\} &= \mathbb{P}\left\{\sup_{x\in\mathcal{X}} \left|\max_{k=1,\ldots,K_n} \alpha_k + \beta_k^T \mathbf{x} - f_0(\mathbf{x})\right| > \varepsilon\right\}, \\ &\leq \mathbb{P}\left\{\max_{i=1,\ldots,n_\delta} \left|\max_{k=1,\ldots,K_n} \alpha_k + \beta_k^T \mathbf{x}_i^\delta - f_0(\mathbf{x}_i^\delta)\right| > \frac{\varepsilon}{\zeta}\right\}, \\ &\leq \mathbb{P}\left\{\max_{i=1,\ldots,n_\delta} \left|\sum_{k=1}^{K_n} \left(\alpha_k + \beta_k^T \mathbf{x}_i^\delta\right) \mathbf{1}_{\{x_i^\delta \in A_k\}} - f_0(\mathbf{x}_i^\delta)\right| > \frac{\varepsilon}{\zeta d_{\mathcal{X}}}\right\}, \\ &< \varepsilon.\end{aligned}$$

■

Appendix B.

The CAP algorithm offers two main computational bottlenecks. First, it searches over all cardinal directions, and only cardinal directions, to produce candidate models. Second, it keeps generating models until no subsets can be split without one having less than the minimum number of observations. In most cases, the optimal number of components is much lower than the terminal number of components.

To alleviate the first problem, we suggest using P' random projections as a basis for search. Using ideas similar to compressive sensing, each projection $\mathbf{g}_j \sim N_p(0,I)$ for $j = 1,\ldots,P'$. Then we search along the direction $\mathbf{g}_j^T\mathbf{x}$ rather than x_j. When we expect the true function to live in a lower dimensional space, as is the case with superfluous covariates, we can set $P' < p$.

We solve the second problem by modifying the stopping rule. Instead of fully growing the tree until each subset has less than $2n/(2\log(n))$ observations, we use generalized cross-validation. We grow the tree until the generalized cross-validation value has increased in two consecutive iterations or each subset has less than $2n/(2\log(n))$ observations. As the generalized cross-validation error is usually concave in K, this heuristic often offers a good fit at a fraction of the computational expense of the full CAP algorithm.

The Fast CAP algorithm has the potential to substantially reduce the $\log(n)^2$ factor by halting the model generation long before K reaches $D\log(n)$. Since every feasible partition is searched for splitting, the computational complexity grows as k gets larger.

The Fast CAP algorithm is summarized as follows.

B.1 Fast Convex Adaptive Partitioning (Fast CAP)

1. **Initialize.** As in CAP.

2. **Split.**

   a. *Generate candidate splits.* Generate candidate model $\hat{M}_{jk\ell}$ by 1) fixing a subset k, 2) generating a random direction j with $\mathbf{g}_j \sim N_p(0,I)$, and 3) dividing the data as follows:

   - set $x_{min}^{jk} = \min\{\mathbf{g}_j^T\mathbf{x}_i : i \in C_k\}$, $x_{max}^{jk} = \max\{\mathbf{g}_j^T\mathbf{x}_i : i \in C_k\}$ and $b_{jk\ell} = a_\ell x_{min}^{jk} + (1-a_\ell)x_{max}^{jk}$
   - set

   $$C_k' = \{i : i \in C_k, \mathbf{g}_j^T\mathbf{x}_i \le b_{jk\ell}\}, \qquad C_{K+1}' = \{i : i \in C_k, \mathbf{g}_j^T\mathbf{x}_i > b_{jk\ell}\},$$
   $$A_k' = \{\mathbf{x} : \mathbf{x} \in A_k, \mathbf{g}_j^T\mathbf{x} \le b_{jk\ell}\}, \qquad A_{K+1}' = \{\mathbf{x} : \mathbf{x} \in A_k, \mathbf{g}_j^T\mathbf{x} > b_{jk\ell}\}.$$

   Then new hyperplanes are fit to each of the new subsets. This is done for L knots, P' dimensions and K subsets.

   b. *Select split.* As in CAP.

3. **Refit.** As in CAP.

4. **Stopping conditions.** Let $GCV(M_K)$ be the generalized cross-validation error for model M_K. Stop if $GCV(M_K) > GCV(M_{K-1})$ and $GCV(M_{K-1}) > GCV(M_{K-2})$ of if $|C_k| < 2n_{min}$ for $k = 1,\ldots,K$. Then select final model as in CAP.

Appendix C.

In this subsection, we empirically examine the effects of the two tunable parameters, the log factor, D, and the number of knots, L. The log factor controls the minimal number of elements in each subset by setting $|C_k| \ge n/(D\log(n))$, and hence it controls the number of subsets, K, at least for large enough n. Increasing D allows the potential accuracy of the estimator to increase, but at the

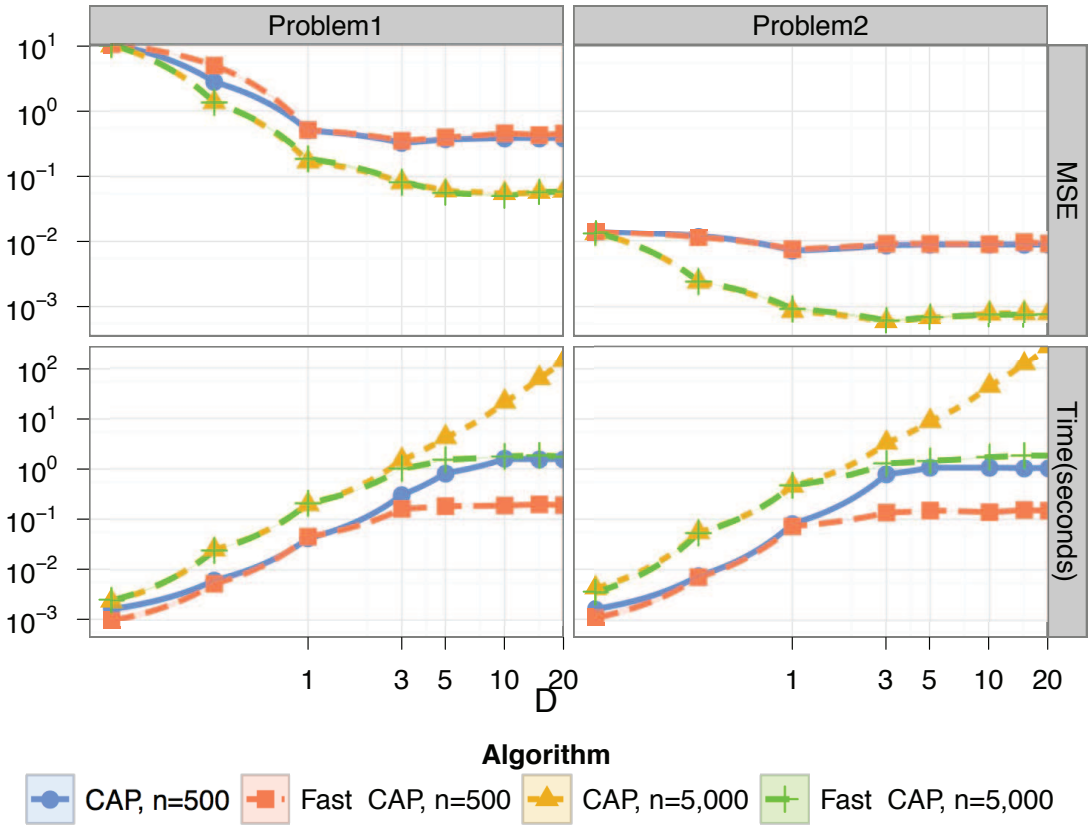


Figure 11: Log factor D (log scale) vs. mean squared error (log scale) for CAP and Fast CAP (top). Log factor D (log scale) vs. runtime in seconds (log scale) (bottom). Both methods were run on problem 1 (left) and problem 2 (right) with $n = 500$ and $n = 5,000$. Lines are mean value and shading represents one standard error.

cost of greater computational time due to the increase in possible values for K and the larger number of possibly admissible sets generated in the splitting step of CAP.

We compared values for D ranging from 0.1 to 20 on problems 1 and 2 with sample sizes of $n = 500$ and $n = 5,000$ over 100 training sets and one testing set. Results are displayed in Figure 11. Note that error may not be strictly decreasing with D because different subsets are proposed under each value. Additionally, Fast CAP is a randomized algorithm so variance in error rate and runtime is to be expected.

Empirically, once $D \geq 1$, there was little substantive error reduction in the models, but the runtime increased as $O(D^2)$ for the full CAP algorithm. Since D controls the maximum partition size, $K_n = D\log(n)$, and a linear regression is fit $K\log(K)$ times, the expected increase in the runtime should only be $O(D\log(D))$. We believe that the extra empirical growth comes from an increased number of feasible candidate splits. In the Fast CAP algorithm, which terminates after generalized cross-validation gains cease to be made, we see runtimes leveling off with higher values of D. Based

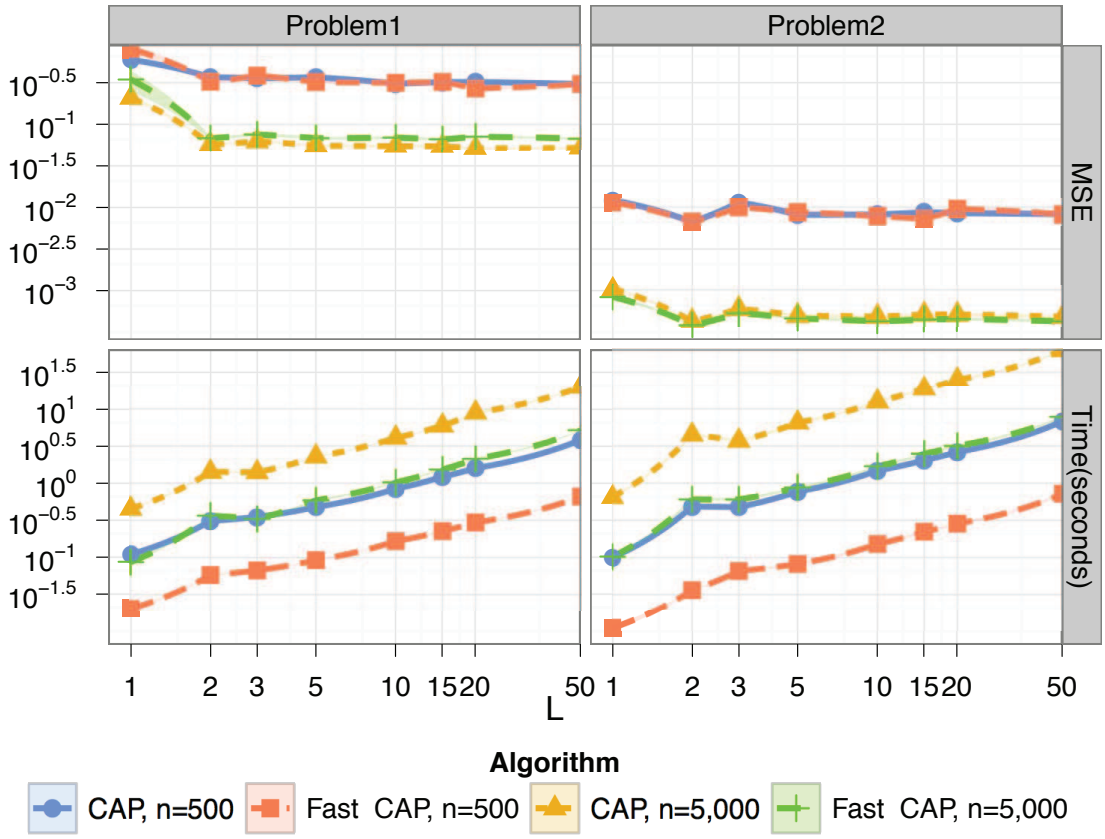


Figure 12: Number of knots L (log scale) vs. mean squared error (log scale) for CAP and Fast CAP (top). Number of knots L (log scale) vs. runtime in seconds (log scale) (bottom). Both methods were run on problem 1 (left) and problem 2 (right) with $n = 500$ and $n = 5,000$. Lines are mean value and shading represents one standard error.

on these results, we believe that setting $D = 3$ offers a good balance between fit and computational expense.

The number of knots, L, determines how many possible subsets will be examined during the splitting step. Like D, an increase in L offers a better fit at the expense of increased computation. We compared values for L ranging from 1 to 50 on problems 1 and 2 with sample sizes of $n = 500$ and $n = 5,000$ over 100 training sets and 1 testing set. Results are displayed in Figure 12.

The changes in fit and runtime are less dramatic with L than they are with D. After $L = 3$, the predictive error rates almost completely stabilized. Runtime increased as $O(L)$ as expected. Due to the minimal increase in computation, we feel that $L = 10$ is a good choice for most settings.

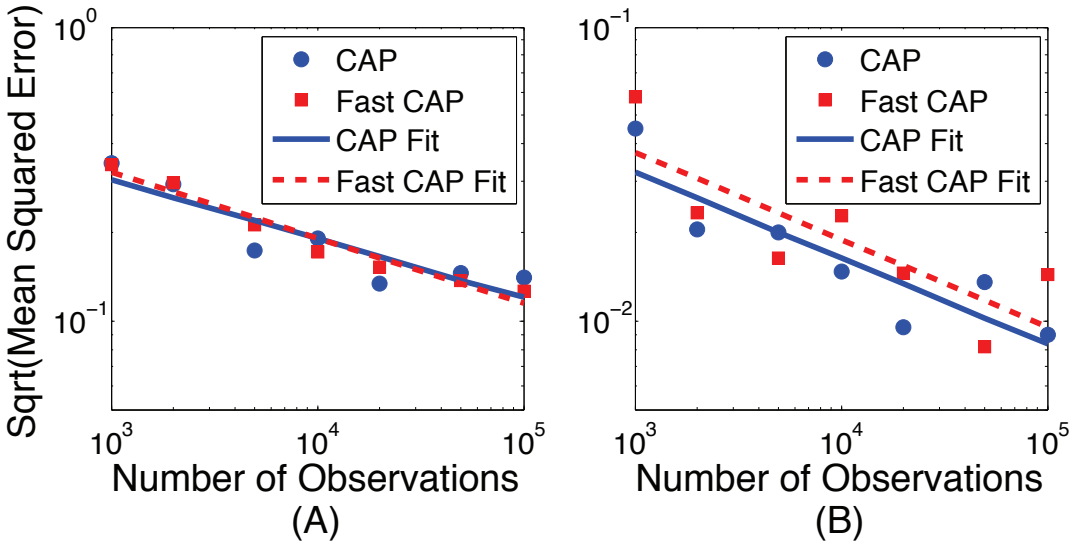


Figure 13: Number of observations n (log scale) vs. square root of mean squared error (log scale) for problem 1 (A) and problem 2 (B). Linear models are fit to find the empirical rate of convergence.

Appendix D.

Although theoretical rates of convergence are not yet available for CAP, we are able to empirically examine them. Rates of convergence for multivariate convex regression have only been studied in two articles of which we are aware. Aguilera et al. (2011) studied rates of convergence for an estimator that is created by first smoothing the data, then evaluating the smoothed data over an ε-net, and finally convexifying the net of smoothed data by taking the convex hull. They showed that the convexify step preserved the rates of the smoothing step. For most smoothing algorithms, these are minimax nonparametric rates, $n^{-\frac{1}{p+2}}$ with respect to the empirical ℓ_2 norm.

Hannah and Dunson (2011) showed adaptive rates for a Bayesian model that places a prior over the set of all piecewise linear functions. Specifically, they showed that if the true mean function f_0 actually maps a d-dimensional linear subspace of $\mathcal{X}$ to $\mathbb{R}$, that is

$$f_0(\mathbf{x}) = g_0(\mathbf{x}\mathbf{A}), \qquad \mathbf{A} \in \mathbb{R}^{p\times d},$$

then their model achieves rates of $\log^{-1}(n)n^{-\frac{1}{d+2}}$ with respect to the empirical ℓ_2 norm. Empirically, we see these types of adaptive rates with CAP.

In Figure 13, we plotted the number of observations against the square root of the mean squared error in a log-log plot for problems 1 and 2. We then fitted a linear model for both CAP and Fast CAP. For problem 1, $p = 5$ but $d = 3$, due to the sum in the quadratic term. Likewise, for problem 2, $p = 10$ but $d = 1$ because it is an exponential of a linear combination. Under standard nonparametric rates, we would expect the slope of the linear model to be $-\frac{1}{7}$ for problem 1 and $-\frac{1}{12}$ for problem 2. However, we see slopes closer to $-\frac{1}{5}$ and $-\frac{1}{3}$ for problems 1 and 2, respectively; values are given

Method	Problem 1	Problem 2
Expected: Rates in p	-0.1429	-0.0833
Expected: Rates in d	-0.2000	-0.3333
Empirical: CAP	-0.2003	-0.2919
Empirical: Fast CAP	-0.2234	-0.2969

Table 2: Slopes for linear models fit to $\log(n)$ vs. $\log(\sqrt{MSE})$ in Figure 13. Expected slopes are given when: 1) rates are with respect to full dimensionality, p, and 2) rates are with respect to dimensionality of linear subspace, d. Empirical slopes are fit to mean squared error generated by CAP and Fast CAP. Note that all empirical slopes are closest to those for linear subspace rates rather than those for full dimensionality rates.

in Table 2. These results strongly imply that CAP achieves adaptive convergence rates of the type shown by Hannah and Dunson (2011) for problems 1 and 2.

References

Néstor Aguilera and Pedro Morin. Approximating optimization problems over convex functions. *Numerische Mathematik*, 111(1):1–34, 2008.

Néstor Aguilera and Pedro Morin. On convex functions and the finite element method. *SIAM Journal on Numerical Analysis*, 47(1):3139–3157, 2009.

Néstor Aguilera, Liliana Forzani, and Pedro Morin. On uniform consistent estimators for convex regression. *Journal of Nonparametric Statistics*, 23(4):897–908, 2011.

Yacine Aït-Sahalia and Jefferson Duarte. Nonparametric option pricing under shape restrictions. *Journal of Econometrics*, 116(1–2):9–47, 2003.

William P. Alexander and Scott D. Grimshaw. Treed regression. *Journal of Computational and Graphical Statistics*, 5(2):156–175, 1996.

Gad Allon, Michael Beenstock, Steven Hackman, Ury Passy, and Alexander Shapiro. Nonparametric estimation of concave production technologies by entropic methods. *Journal of Applied Econometrics*, 22(4):795–816, 2007.

Herman J. Bierens and Donna K. Ginther. Integrated conditional moment testing of quantile regression models. *Empirical Economics*, 26(1):307–324, 2001.

Melanie Birke and Holger Dette. Estimating a convex function in nonparametric regression. *Scandinavian Journal of Statistics*, 34(2):384–404, 2007.

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

Stephen Boyd, Seung-Jean Kim, Lieven Vandenberghe, and Arash Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, 2007.

Leo Breiman. Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, 39(3):999–1013, 1993.

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

Leo Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3): 229–242, 2000.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Chapman & Hall/CRC, Boca Raton, Florida, 1984.

Hugh D. Brunk. Maximum likelihood estimates of monotone parameters. *The Annals of Mathematical Statistics*, 26(4):607–616, 1955.

Jacques F. Carriere. Valuation of the early-exercise price for options using simulations and nonparametric regression. *Insurance: Mathematics and Economics*, 19(1):19–30, 1996.

I-Shou Chang, Li-Chu Chien, Chao A. Hsiung, Chi-Chung Wen, and Yuh-Jenn Wu. Shape restricted regression with random Bernstein polynomials. *IMS Lecture Notes-Monograph Series*, 54:187–202, 2007.

Probal Chaudhuri, Min-Ching Huang, Wei-Yin Loh, and Ruji Yao. Piecewise-polynomial regression trees. *Statistica Sinica*, 4(1):143–167, 1994.

Probal Chaudhuri, Wen-Da Lo, Wei-Yin Loh, and Ching-Ching Yang. Generalized regression trees. *Statistica Sinica*, 5(1):641–666, 1995.

Madeleine Cule and Richard Samworth. Theoretical properties of the log-concave maximum likelihood estimator of a multidimensional density. *Electronic Journal of Statistics*, 4:254–270, 2010.

Madeleine Cule, Richard Samworth, and Michael Stewart. Maximum likelihood estimation of a multi-dimensional log-concave density. *Journal of the Royal Statistical Society, Series B*, 72(5): 545–607, 2010.

Warren Dent. A note on least squares fitting of functions constrained to be either nonnegative, nondecreasing or convex. *Management Science*, 20(1):130–132, 1973.

Alin Dobra and Johannes Gehrke. SECRET: a scalable linear regression tree algorithm. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 481–487, Berlin, Germany, 2002. ACM.

Richard L. Dykstra. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842, 1983.

Donald A. S. Fraser and Hélène Massam. A mixed primal-dual bases algorithm for regression under inequality constraints. Application to concave regression. *Scandinavian Journal of Statistics*, 16 (1):65–74, 1989.

Jerome H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1): 1–141, 1991.

Paul Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer Verlag, New York, New York, 2004.

Gene H. Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.0 beta. `http://cvxr.com/cvx`, September 2012.

Piet Groeneboom, Geurt Jongbloed, and Jon A. Wellner. Estimation of a convex function: characterizations and asymptotic theory. *Annals of Statistics*, 29(6):1653–1698, 2001.

László Györfi, Michael Kohler, Adam Krzyżak, and Harro Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, New York, New York, 2002.

Peter Hall and Li-Shan Huang. Nonparametric kernel regression subject to monotonicity constraints. *The Annals of Statistics*, 29(3):624–647, 2001.

Lauren A. Hannah and David B. Dunson. Bayesian nonparametric multivariate convex regression. Technical Report arXiv:1109.0322v1, Department of Statistical Science, Duke University, Durham, North Carolina, 2011.

Lauren A. Hannah and David B. Dunson. Ensemble methods for convex regression with applications to geometric programming based circuit design. In *Proceedings of the 29th Annual International Conference on Machine Learning*, Edinburgh, United Kingdom, 2012.

D. L. Hanson and Gordon Pledger. Consistency in concave regression. *The Annals of Statistics*, 4 (6):1038–1050, 1976.

Martin B. Haugh and Leonid Kogan. Pricing American options: a duality approach. *Operations Research*, 52(2):258–270, 2004.

Daniel J. Henderson and Christopher F. Parmeter. Imposing economic constraints in nonparametric regression: survey, implementation and extension. In Q. Li and J. S. Racine, editors, *Nonparametric Econometric Methods (Advances in Econometrics)*, volume 25, pages 433–469. Emerald Publishing Group Limited, Bingley, United Kingdom, 2009.

Clifford Hildreth. Point estimates of ordinates of concave functions. *Journal of the American Statistical Association*, 49(267):598–619, 1954.

Charles A. Holloway. On the estimation of convex functions. *Operations Research*, 27(2):401–407, 1979.

Arezou Keshavarz, Yang Wang, and Stephen Boyd. Imputing a convex objective function. In *Proceedings of the IEEE International Symposium on Intelligent Control*, pages 613–619, Denver, Colorado, 2011.

Jintae Kim, Jaeseo Lee, Lieven Vandenberghe, and C.K.-Ken Yang. Techniques for improving the accuracy of geometric-programming based analog circuit design optimization. In *Proceedings of the IEEE International Conference on Computer Aided Design*, pages 863–870, San Jose, California, 2004.

Farinaz Koushanfar, Mehrdad Majzoobi, and Miodrag Potkonjak. Nonparametric combinatorial regression for shape constrained modeling. *IEEE Transactions on Signal Processing*, 58(2):626–637, 2010.

Timo Kuosmanen. Representation theorem for convex nonparametric least squares. *Econometrics Journal*, 11(2):308–325, 2008.

Timo Kuosmanen and Andrew L. Johnson. Data envelopment analysis as nonparametric least-squares regression. *Operations Research*, 58(1):149–160, 2010.

Eunji Lim. Response surface computation via simulation in the presence of convexity constraints. In *Proceedings of the 2010 Winter Simulation Conference*, pages 1246–1254, Baltimore, Maryland, 2010.

Eunji Lim and Peter W. Glynn. Consistency of multi-dimensional convex regression. *Operations Research*, 60(1):196–208, 2012.

Francis A. Longstaff and Eduardo S. Schwartz. Valuing American options by simulation: a simple least-squares approach. *Review of Financial Studies*, 14(1):113–147, 2001.

Alessandro Magnani and Stephen Boyd. Convex piecewise-linear fitting. *Optimization and Engineering*, 10(1):1–17, 2009.

Enno Mammen. Nonparametric regression under qualitative smoothness assumptions. *The Annals of Statistics*, 19(2):741–759, 1991.

Mary C. Meyer. Inference using shape-restricted regression splines. *Annals of Applied Statistics*, 2 (3):1013–1033, 2008.

Mary C. Meyer, Amber J. Hackstadt, and Jennifer A. Hoeting. Bayesian estimation and inference for generalised partial linear models using shape-restricted splines. *Journal of Nonparametric Statistics*, 23(4):867–884, 2011.

Renato D. C. Monteiro and Ilan Adler. Interior path following primal-dual algorithms. Part II: convex quadratic programming. *Mathematical Programming*, 44(1–3):43–66, 1989.

Brian Neelon and David B. Dunson. Bayesian isotonic regression and trend analysis. *Biometrics*, 60(2):398–406, 2004.

Andrew Nobel. Histogram regression estimation using data-dependent partitions. *The Annals of Statistics*, 24(3):1084–1105, 1996.

Duncan Potts and Claude Sammut. Incremental learning of linear model trees. *Machine Learning*, 61(1):5–48, 2005.

Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley-Interscience, Hoboken, New Jersey, 2007.

J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1993.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, Massachusetts, 2006.

Sanghamitra Roy, Weijen Chen, Charlie Chung-Ping Chen, and Yu Hen Hu. Numerically convex forms and their application in gate sizing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(9):1637–1647, 2007.

Dominic Schuhmacher and Lutz Dümbgen. Consistency of multivariate log-concave density estimators. *Statistics & Probability Letters*, 80(5-6):376–380, 2010.

Emilio Seijo and Bodhisattva Sen. Nonparametric least squares estimation of a multivariate convex regression function. *The Annals of Statistics*, 39(3):1580–1607, 2011.

Thomas S. Shively, Thomqw W. Sager, and Stephen G. Walker. A Bayesian approach to nonparametric monotone function estimation. *Journal of the Royal Statistical Society, Series B*, 71 (1):159–175, 2009.

Thomas S. Shively, Stephen G. Walker, and Paul Damien. Nonparametric function estimation subject to monotonicity, convexity and other shape constraints. *Journal of Econometrics*, 161(2): 166–181, 2011.

Huseyin Topaloglu and Warren B. Powell. An algorithm for approximating piecewise linear concave functions from sample gradients. *Operations Research Letters*, 31(1):66–76, 2003.

Alejandro Toriello, George Nemhauser, and Martin Savelsbergh. Decomposing inventory routing problems with approximate value functions. *Naval Research Logistics*, 57(8):718–727, 2010.

John N. Tsitsiklis and Benjamin Van Roy. Optimal stopping of Markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *IEEE Transactions on Automatic Control*, 44(10):1840–1851, 1999.

John N. Tsitsiklis and Benjamin Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.

Berwin A. Turlach. Shape constrained smoothing using smoothing splines. *Computational Statistics*, 20(1):81–103, 2005.

Hal R. Varian. The nonparametric approach to demand analysis. *Econometrica*, 50(4):945–973, 1982.

Hal R. Varian. The nonparametric approach to production analysis. *Econometrica*, 52(3):579–597, 1984.

Yongqiao Wang and He Ni. Multivariate convex support vector regression with semidefinite programming. *Knowledge-Based Systems*, 30:87–94, 2012.

C. F. Jeff Wu. Some algorithms for concave and isotonic regression. In S. H. Zanakis and J. S. Rustagi, editors, *Studies in the Management Sciences*, volume 19, pages 105–116. North-Holland, Amsterdam, 1982.

Journal of Machine Learning Research 14 (2013) 3295-3320 Submitted 9/12; Revised 7/13; Published 11/13

Fast MCMC Sampling for Markov Jump Processes and Extensions

Vinayak Rao[*] VRAO@GATSBY.UCL.AC.UK
Department of Statistical Science
Duke University
Durham, NC, 27708-0251, USA

Yee Whye Teh Y.W.TEH@STATS.OX.AC.UK
Department of Statistics
1 South Parks Road
Oxford OX1 3TG, UK

Editor: Christopher Meek

Abstract

Markov jump processes (or continuous-time Markov chains) are a simple and important class of continuous-time dynamical systems. In this paper, we tackle the problem of simulating from the posterior distribution over paths in these models, given partial and noisy observations. Our approach is an auxiliary variable Gibbs sampler, and is based on the idea of *uniformization*. This sets up a Markov chain over paths by alternately sampling a finite set of virtual jump times given the current path, and then sampling a new path given the set of extant and virtual jump times. The first step involves simulating a piecewise-constant inhomogeneous Poisson process, while for the second, we use a standard hidden Markov model forward filtering-backward sampling algorithm. Our method is exact and does not involve approximations like time-discretization. We demonstrate how our sampler extends naturally to MJP-based models like Markov-modulated Poisson processes and continuous-time Bayesian networks, and show significant computational benefits over state-of-the-art MCMC samplers for these models.

Keywords: Markov jump process, MCMC, Gibbs sampler, uniformization, Markov-modulated Poisson process, continuous-time Bayesian network

1. Introduction

The Markov jump process (MJP) extends the discrete-time Markov chain to continuous time, and forms a simple and popular class of continuous-time dynamical systems. In Bayesian modelling applications, the MJP is widely used as a prior distribution over the piecewise-constant evolution of the state of a system. The Markov property of the MJP makes it both a realistic model for various physical and chemical systems, as well as a convenient approximation for more complex phenomena in biology, finance, queueing systems etc. In chemistry and biology, stochastic kinetic models use the state of an MJP to represent the sizes of various interacting *species* (e.g., Gillespie, 1977; Golightly and Wilkinson, 2011). In queueing applications, the state may represent the number of pending jobs in a queue (Breuer, 2003; Tijms, 1986), with the arrival and processing of jobs treated as memoryless events. MJPs find wide application in genetics, for example, an MJP trajectory is sometimes used to represent a segmentation of a strand of genetic matter. Here 'time' represents

∗. Corresponding author

position along the strand, with particular motifs occurring with different rates in different regions (Fearnhead and Sherlock, 2006). MJPs are also widely used in finance, for example, Elliott and Osakwe (2006) use an MJP to model switches in the parameters that govern the dynamics of stock prices (the latter being modelled with a Lévy process).

In the Bayesian setting, the challenge is to characterize the posterior distribution over MJP trajectories given noisy observations; this typically cannot be performed analytically. Various sampling-based (Fearnhead and Sherlock, 2006; Boys et al., 2008; El-Hay et al., 2008; Fan and Shelton, 2008; Hobolth and Stone, 2009) and deterministic (Nodelman et al., 2002, 2005; Opper and Sanguinetti, 2007; Cohn et al., 2010) approximations have been proposed in the literature, but come with problems: they are often generic methods that do not exploit the structure of the MJP, and when they do, involve expensive computations like matrix exponentiation, matrix diagonalization or root-finding, or are biased, involving some form of time-discretization or independence assumptions. Moreover, these methods do not extend easily to more complicated likelihood functions which require specialized algorithms (for instance, the contribution of Fearnhead and Sherlock (2006) is to develop an exact sampler for Markov-modulated Poisson processes (MMPPs), where an MJP modulates the rate of a Poisson process).

In this work, an extension of Rao and Teh (2011a), we describe a novel Markov chain Monte Carlo (MCMC) sampling algorithm for MJPs that avoids the need for the expensive computations described previously, and does not involve any form of approximation (i.e., our MCMC sampler converges to the true posterior). Importantly, our sampler is easily adapted to complicated extensions of MJPs such as MMPPs and continuous-time Bayesian networks (CTBNs) (Nodelman et al., 2002), and is significantly more efficient than the specialized samplers developed for these models. Like many existing methods, our sampler introduces auxiliary variables which simplify the structure of the MJP, using an idea called *uniformization*. Importantly, unlike some existing methods which produce *independent* posterior samples of these auxiliary variables, our method samples these *conditioned* on the current sample trajectory. While the former approach depends on the observation process, and can be hard for complicated likelihood functions, ours results in a simple distribution over the auxiliary variables that is independent of the observations. The observations are accounted for during a straightforward discrete-time forward-filtering backward-sampling step to resample a new trajectory. The overall structure of our algorithm is that of an auxiliary variable Gibbs sampler, alternately resampling the auxiliary variables given the MJP trajectory, and the trajectory given the auxiliary variables.

In Section 2 we briefly review Markov jump processes. In Section 3 we introduce the idea of uniformization and describe our MCMC sampler for the simple case of a discretely observed MJP. In Section 4, we apply our sampler to the Markov-modulated Poisson process, while in Section 5, we describe continuous-time Bayesian networks, and extend our algorithm to that setting. In both sections, we report experiments comparing our algorithm to state-of-the-art sampling algorithms developed for these models. We end with a discussion in Section 6.

2. Markov Jump Processes (MJPs)

A Markov jump process $(\mathbf{S}(t),\ t \in \mathbb{R}_+)$ is a stochastic process with right-continuous, piecewise-constant paths (see for example Çinlar, 1975). The paths themselves take values in some countable space $(\mathcal{S},\Sigma_{\mathcal{S}})$, where $\Sigma_{\mathcal{S}}$ is the discrete σ-algebra. As in typical applications, we assume $\mathcal{S}$ is finite (say $\mathcal{S} = \{1,2,...N\}$). We also assume the process is homogeneous, implying (together with the

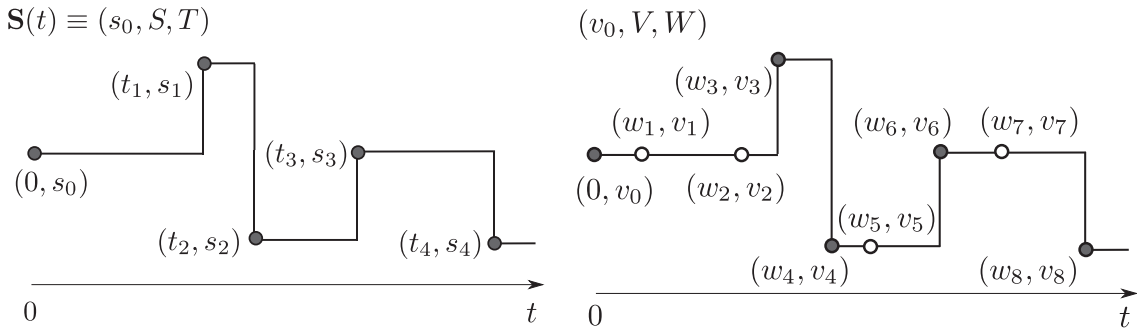


Figure 1: (left) An MJP path (s_0, S, T), (right) a uniformized representation (v_0, V, W).

Markov property) that for all times $t, t' \in \mathbb{R}_+$ and states $s, s' \in \mathcal{S}$,

$$p\left(\mathbf{S}(t'+t) = s | \mathbf{S}(t') = s', (\mathbf{S}(u), u < t')\right) = [P_t]_{ss'}$$

for some stochastic matrix P_t that depends only on t. The family of transition matrices $(P_t,\ t \geq 0)$ is defined by a matrix $A \in \mathbb{R}^{N \times N}$ called the *rate matrix* or *generator* of the MJP. A is the time-derivative of P_t at $t = 0$, with

$$P_t = \exp(At), \tag{1}$$
$$p(\mathbf{S}(t' + \mathrm{d}t) = s | \mathbf{S}(t') = s') = A_{ss'}\mathrm{d}t \qquad (\text{for } s \neq s'),$$

where Equation (1) is the matrix exponential. The off-diagonal elements of A are nonnegative, and represent the rates of transiting from one state to another. Its diagonal entries are $A_s \equiv A_{ss} = -\sum_{s' \neq s} A_{s's}$ for each s, so that its columns sum to 0, with $-A_s = |A_s|$ characterizing the total rate of leaving state s.

Consider a time interval $\mathcal{T} \equiv [t_{start}, t_{end}]$, with the Borel σ-algebra $\Sigma_{\mathcal{T}}$. Let π_0 be a density with respect to the counting measure $\mu_{\mathcal{S}}$ on $(\mathcal{S}, \Sigma_{\mathcal{S}})$; this defines the initial distribution over states at t_{start}. Then an MJP is described by the following generative process over paths on this interval, commonly called *Gillespie's algorithm* (Gillespie, 1977):

Algorithm 1 Gillespie's algorithm to sample an MJP path on the interval $[t_{start}, t_{end}]$

Input: The rate matrix A and the initial distribution over states π_0.
Output: An MJP trajectory $\mathbf{S}(t) \equiv (s_0, S, T)$.

1: Assign the MJP a state $s_0 \sim \pi_0$. Set $t_0 = t_{start}$ and $i = 0$.
2: **loop**
3: Draw $z \sim \exp(|A_{s_i}|)$.
4: **If** $t_i + z > t_{end}$ **then return** $(s_0, \ldots, s_i, t_1, \ldots, t_i)$ and **stop**.
5: Increment i and let $t_i = t_{i-1} + z$.
6: The MJP jumps to a new state $s_i = s$ at time t_i, for an $s \neq s_{i-1}$,
7: with probability proportional to $A_{ss_{i-1}}$.
8: **end loop**

If all event rates are finite, an MJP trajectory will almost surely have only a finite number of jumps. Let there be n jumps, and let these occur at the ordered times $(t_1, \cdots, t_n)$. Define $T \equiv (t_1, \cdots, t_n)$, and let $S \equiv (s_1, \cdots, s_n)$ be the corresponding sequence of states, where $s_i = \mathbf{S}(t_i)$. The triplet (s_0, S, T) completely characterizes the MJP trajectory over $\mathcal{T}$ (Figure 1 (left)).

From Gillespie's algorithm, we see that sampling an MJP trajectory involves sequentially sampling $n+1$ waiting times from exponential densities with one of N rates, and n new states from one of N discrete distributions, each depending on the previous state. The ith waiting time equals $(t_i - t_{i-1})$ and is drawn from an exponential with rate $|A_{s_{i-1}}|$, while the probability the ith state equals s_i is $A_{s_i s_{i-1}}/|A_{s_{i-1}}|$. The last waiting time can take any value greater than $t_{end} - t_n$. Thus, under an MJP, a random element (s_0, S, T) has density

$$\begin{aligned} p(s_0, S, T) &= \pi_0(s_0) \left(\prod_{i=1}^{n} |A_{s_{i-1}}| e^{-|A_{s_{i-1}}|(t_i - t_{i-1})} \frac{A_{s_i s_{i-1}}}{|A_{s_{i-1}}|} \right) \cdot e^{-|A_{s_n}|(t_{end} - t_n)} \\ &= \pi_0(s_0) \left(\prod_{i=1}^{n} A_{s_i s_{i-1}} \right) \exp\left(- \int_{t_{start}}^{t_{end}} |A_{\mathbf{S}(t)}| \mathrm{d}t \right). \end{aligned} \tag{2}$$

To be precise, we must state the base measure with respect to which the density above is defined. The reader might wish to skip these details (and for more details, we recommend Daley and Vere-Jones, 2008). Let $\mu_{\mathcal{T}}$ be Lebesgue measure on $\mathcal{T}$. Recalling that the state space of the MJP is $\mathcal{S}$, we can view (S, T) as a sequence of elements in the product space $\mathcal{M} \equiv \mathcal{S} \times \mathcal{T}$. Let $\Sigma_{\mathcal{M}}$ and $\mu_{\mathcal{M}} = \mu_{\mathcal{S}} \times \mu_{\mathcal{T}}$ be the corresponding product σ-algebra and product measure. Define $\mathcal{M}^n$ as the n-fold product space with the usual product σ-algebra $\Sigma^n_{\mathcal{M}}$ and product measure $\mu^n_{\mathcal{M}}$. Now let $\mathcal{M}^{\cup} \equiv \bigcup_{i=0}^{\infty} \mathcal{M}^i$ be a union space, elements of which represent finite length pure-jump paths.[1] Let $\Sigma^{\cup}_{\mathcal{M}}$ be the corresponding union σ-algebra, where each measurable set $B \in \Sigma^{\cup}_{\mathcal{M}}$ can be expressed as $B = \cup_{i=0}^{\infty} B^i$ with $B^i = B \cap \mathcal{M}^i \in \Sigma^i_{\mathcal{M}}$. Assign this space the measure $\mu^{\cup}_{\mathcal{M}}$ defined as:

$$\mu^{\cup}_{\mathcal{M}}(B) = \sum_{i=0}^{\infty} \mu^i_{\mathcal{M}}(B^i).$$

Then, any element $(s_0, S, T) \in \mathcal{S} \times \mathcal{M}^{\cup}$ sampled from Gillespie's algorithm has density w.r.t. $\mu_{\mathcal{S}} \times \mu^{\cup}_{\mathcal{M}}$ given by Equation (2).

3. MCMC Inference via Uniformization

In this paper, we are concerned with the problem of sampling MJP paths over the interval $\mathcal{T} \equiv [t_{start}, t_{end}]$ given noisy observations of the state of the MJP. In the simplest case, we observe the process at the boundaries t_{start} and t_{end}. More generally, we are given the initial distribution over states π_0 as well as a set of O noisy observations $X = \{X_{t^o_1}, \ldots X_{t^o_O}\}$ at times $T^o = \{t^o_1, \ldots, t^o_O\}$ with likelihoods $p(X_{t^o_i} | \mathbf{S}(t^o_i))$, and we wish to sample from the posterior $p(s_0, S, T | X)$. Here we have implicitly assumed that the observation times T^o are fixed. Sometimes the observation times themselves can depend on the state of the MJP, resulting effectively in *continuous-time* observations. This is the case for the Markov-modulated Poisson process and CTBNs. As we will show later, our method handles these cases quite naturally as well.

1. Define $\mathcal{M}^0$ as a point satisfying $\mathcal{M}^0 \times \mathcal{M} = \mathcal{M} \times \mathcal{M}^0 = \mathcal{M}$ (Daley and Vere-Jones, 2008).

A simple approach to inference is to discretize time and work with the resulting approximation. The time-discretized MJP corresponds to the familiar discrete-time Markov chain, and its Markov structure can be exploited to construct dynamic programming algorithms like the forward-filtering backward-sampling (FFBS) algorithm (Früwirth-Schnatter, 1994; Carter and Kohn, 1996; see also Appendix A) to sample posterior trajectories efficiently. However, time-discretization introduces a bias into our inferences, as the system can change state only at a fixed set of times, and as the maximum number of state changes is limited to a finite number. To control this bias, one needs to discretize time at a fine granularity, resulting in long Markov chains, and expensive computations.

Recently, there has been growing interest in constructing *exact* MCMC samplers for MJPs without any approximations such as time-discretization. We review these in Section 3.3. One class of methods exploits the fact an MJP can be exactly represented by a discrete-time Markov chain on a *random* time-discretization. Unlike discretization on a regular grid, a random grid can be quite coarse without introducing any bias. Given this discretization, we can use the FFBS algorithm to perform efficient sampling. However, we do not observe the random discretization, and thus also need to sample this from its posterior distribution. This posterior now depends on the likelihood process, and a number of algorithms attempt to solve this problem for specific observation processes. Our approach is to resample the discretization conditioned on the system trajectory. As we will see this is *independent* of the likelihood process, resulting in a simple, flexible and efficient MCMC sampler.

3.1 Uniformization

We first introduce the idea of *uniformization* (Jensen, 1953; Çinlar, 1975; Hobolth and Stone, 2009), which forms the basis of our sampling algorithm. For an MJP with rate-matrix A, choose some $\Omega \geq \max_s |A_s|$. Let $W = (w_1, \ldots, w_{|W|})$ be an ordered set of times on the interval $[t_{start}, t_{end}]$ drawn from a homogeneous Poisson process with intensity Ω. W constitutes a random discretization of the time-interval $[t_{start}, t_{end}]$.

Next, letting I be the identity matrix, observe that $B = \left(I + \frac{1}{\Omega}A\right)$ is a stochastic matrix (it has nonnegative elements, and its columns sum to one). Run a discrete-time Markov chain with initial distribution π_0 and transition matrix B on the times in W; this is a Markov chain *subordinated* to the Poisson process W. The Markov chain will assign a set of states (v_0, V); v_0 at the initial time t_{start}, and $V = (v_1, \ldots, v_{|V|})$ at the discretization times W (so that $|V| = |W|$). In particular, v_0 is drawn from π_0, while v_i is drawn from the probability vector given by the v_{i-1}th column of B. Just as (s_0, S, T) characterizes an MJP path, (v_0, V, W) also characterizes a sample path of some piecewise-constant, right-continuous stochastic process on $[t_{start}, t_{end}]$. Observe that the matrix B allows self-transitions, so that unlike S, V can jump from a state back to the same state. We treat these as *virtual* jumps, and regard (v_0, V, W) as a redundant representation of a pure-jump process that always jumps to a new state (see Figure 1 (right)). The virtual jumps provide a mechanism to 'thin' the set W, thereby rejecting some of its events. This corrects for the fact that since the Poisson rate Ω dominates the leaving rates of all states of the MJP, W will on average contain more events than there are jumps in the MJP path. As the parameter Ω increases, the number of events in W increases; at the same time the diagonal entries of B start to dominate, so that the number of self-transitions (thinned events) also increases. The next proposition shows that these two effects exactly compensate each other, so that the process characterized by (v_0, V, W) is precisely the desired MJP.

Proposition 1 *(Jensen, 1953) For any* $\Omega \geq \max_s |A_s|$, (s_0, S, T) *and* (v_0, V, W) *define the same Markov jump process* $\mathbf{S}(t)$.

Proof We follow Hobolth and Stone (2009). From Equation (1), the marginal distribution of the MJP at time t is given by

$$\begin{aligned}\pi_t &= \exp(At)\pi_0 \\ &= \exp\left(\Omega(B-I)t\right)\pi_0 \\ &= \exp\left(-\Omega t\right)\exp\left(\Omega t B\right)\pi_0 \\ &= \sum_{n=0}^{\infty}\left(\left(\exp\left(-\Omega t\right)\frac{(\Omega t)^n}{n!}\right)\left(B^n\pi_0\right)\right).\end{aligned}$$

The first term in the summation is the probability that a rate Ω Poisson produces n events in an interval of length t, that is, that $|W| = n$. The second term gives the marginal distribution over states for a discrete-time Markov chain after n steps, given that the initial state is drawn from π_0, and subsequent states are assigned according to a transition matrix B. Summing over n, we obtain the marginal distribution over states at time t. Since the transition kernels induced by the uniformization procedure agree with those of the Markov jump process ($\exp(At)$) for all t, and since the two processes also share the same initial distribution of states, π_0, all finite dimensional distributions agree. Following Kolmogorov's extension theorem (Kallenberg, 2002), both define versions of the same stochastic process.

■

A more direct but cumbersome approach is note that (v_0, V, W) is also an element of the space $\mathcal{S} \times \mathcal{M}^{\cup}$. We can then write down its density $p(v_0, V, W)$ w.r.t. $\mu_{\mathcal{S}} \times \mu^{\cup}$, and show that marginalizing out the number and locations of self-transitions recovers Equation (2). While we do not do this, we will derive the density $p(v_0, V, W)$ for later use. As in Section 2, let $\mathcal{T}^{\cup}$ and $\mathcal{S}^{\cup}$ denote the measure spaces consisting of finite sequences of times and states respectively, and let $\mu^{\cup}_{\mathcal{T}}$ and $\mu^{\cup}_{\mathcal{S}}$ be the corresponding base measures. The Poisson realization W is determined by waiting times sampled from a rate Ω exponential distribution, so that following Equation (2), W has density w.r.t. $\mu^{\cup}_{\mathcal{T}}$ given by

$$p(W) = \Omega^{|W|} e^{-\Omega(t_{end} - t_{start})}. \tag{3}$$

Similarly, from the construction of the Markov chain, it follows that the state assignment (v_0, V) has probability density w.r.t. $\mu_{\mathcal{S}} \times \mu^{\cup}_{\mathcal{S}}$ given by

$$p(v_0, V|W) = \pi_0(v_0)\prod_{i=1}^{|V|}\left(1 + \frac{A_{v_i}}{\Omega}\right)^{1(v_i = v_{i-1})}\left(\frac{A_{v_i v_{i-1}}}{\Omega}\right)^{1(v_i \neq v_{i-1})}.$$

Since under uniformization $|V| = |W|$, it follows that

$$\begin{aligned}\mu^{\cup}_{\mathcal{S}}(\mathrm{d}V) \times \mu^{\cup}_{\mathcal{T}}(\mathrm{d}W) &= \mu^{|V|}_{\mathcal{S}}(\mathrm{d}V) \times \mu^{|W|}_{\mathcal{T}}(\mathrm{d}W) \\ &= \left(\mu_{\mathcal{T}} \times \mu_{\mathcal{S}}\right)^{|V|}(\mathrm{d}(V, W)) \\ &= \mu^{\cup}_{\mathcal{M}}(\mathrm{d}(V, W)).\end{aligned} \tag{4}$$

Thus, from Equations (3) and (4), (v_0, V, W) has density w.r.t. $\mu_{\mathcal{S}} \times \mu_{\mathcal{M}}^{\cup}$ given by

$$p(v_0, V, W) = e^{-\Omega(t_{end} - t_{start})} \pi_0(v_0) \prod_{i=1}^{|V|} (\Omega + A_{v_i})^{1(v_i = v_{i-1})} \left(A_{v_i v_{i-1}}\right)^{1(v_i \neq v_{i-1})}. \tag{5}$$

3.2 The MCMC Algorithm

We adapt the uniformization scheme described above to construct an auxiliary variable Gibbs sampler. Recall that the only difference between (s_0, S, T) and (v_0, V, W) is the presence of an auxiliary set of virtual jumps in the latter. Call the virtual jump times $U_{\mathcal{T}}$; associated with $U_{\mathcal{T}}$ is a sequence of states $U_{\mathcal{S}}$. $U_{\mathcal{S}}$ is uniquely determined by (s_0, S, T) and $U_{\mathcal{T}}$ (see Figure 1(right)), and we say this configuration is *compatible*. Let $U = (U_{\mathcal{S}}, U_{\mathcal{T}})$, and observe that for compatible values of $U_{\mathcal{S}}$, (s_0, S, T, U) and (v_0, V, W) represent the same point in $\mathcal{S} \times \mathcal{M}^{\cup}$.

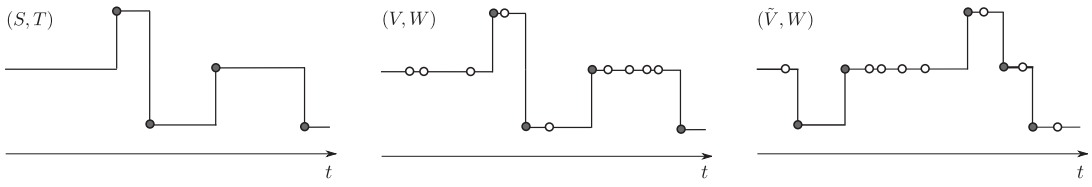


Figure 2: Uniformization-based Gibbs sampler: starting with an MJP trajectory (left), resample the thinned events (middle) and then resample the trajectory given all Poisson events (right). Discard the thinned events and repeat.

Given an MJP trajectory (s_0, S, T) (Figure 2 (left)), we proceed by first sampling the set of virtual jumps $U_{\mathcal{T}}$ given (s_0, S, T), as a result recovering the uniformized characterization (s_0, V, W) (Figure 2 (middle)). This corresponds to a random discretization of $[t_{start}, t_{end}]$ at times W. We now discard the state sequence V, and perform a simple HMM forward-filtering backward-sampling step to resample a new state sequence $\tilde{V}$. Finally, dropping the virtual jumps in $(s_0, \tilde{V}, W)$ gives a new MJP path $(s_0, \tilde{S}, \tilde{T})$. Figure 2 describes an iteration of the MCMC algorithm.

The next proposition shows that conditioned on (s_0, S, T), the virtual jump times $U_{\mathcal{T}}$ are distributed as an *inhomogeneous* Poisson process with intensity $R(t) = \Omega + A_{\mathbf{S}(t)}$ (we remind the reader that A has a negative diagonal, so that $R(t) \leq \Omega$). This intensity is piecewise-constant, taking the value $r_i = \Omega + A_{s_i}$ on the interval $[t_i, t_{i+1})$ (with $t_0 = t_{start}$ and $t_{n+1} = t_{end}$), so it is easy to sample $U_{\mathcal{T}}$ and thus U.

Proposition 2 *For any $\Omega \geq \max_s(|A_s|)$, both (s_0, S, T, U) and (v_0, V, W) have the same density w.r.t. $\mu_{\mathcal{S}} \times \mu_{\mathcal{M}}^{\cup}$. In other words, the Markov jump process (s_0, S, T) along with virtual jumps U drawn from the inhomogeneous Poisson process as above is equivalent to the times W being drawn from a Poisson process with rate Ω, followed by the states (v_0, V) being drawn from the subordinated Markov chain.*

Proof Let $n = |T|$ be the number of jumps in the current MJP trajectory. Define $|U_i|$ as the number of auxiliary times in interval (t_i, t_{i+1}). Then, $|U_{\mathcal{T}}| = \sum_{i=0}^{n} |U_i|$. If $U_{\mathcal{T}}$ is sampled from a piecewise-constant inhomogeneous Poisson process, its density is the product of the densities of a sequence of

homogeneous Poisson processes, and from Equation (3) is

$$p(U_{\mathcal{T}}|s_0,S,T) = \left(\prod_{i=0}^{n}(\Omega+A_{s_i})^{|U_i|}\right)\exp\left(-\int_{t_{start}}^{t_{end}}(\Omega+A_{\mathbf{S}(t)})dt\right) \tag{6}$$

w.r.t. $\mu_{\mathcal{T}}^{\cup}$. Having realized the times $U_{\mathcal{T}}$, the associated states $U_{\mathcal{S}}$ are determined too (elements of $U_{\mathcal{S}}$ in the interval (t_{i-1},t_i) equal s_{i-1}). Thus $U=(U_{\mathcal{S}},U_{\mathcal{T}})$ given (s_0,S,T) has the same density as Equation (6), but now w.r.t. $\mu_{\mathcal{M}}^{\cup}$, and now restricted to elements of $\mathcal{M}^{\cup}$ where $U_{\mathcal{S}}$ is compatible with $(S,T,U_{\mathcal{T}})$. Multiplying Equations (2) and (6), we see that (s_0,S,T,U) has density

$$p(s_0,S,T,U) = e^{-\Omega(t_{end}-t_{start})}\pi_0(s_0)\prod_{i=0}^{n}(\Omega+A_{s_i})^{|U_i|}\prod_{i=1}^{n}A_{s_is_{i-1}}$$

w.r.t. $\mu_{\mathcal{S}}\times\mu_{\mathcal{M}}^{\cup}\times\mu_{\mathcal{M}}^{\cup}$. However, by definition,

$$\begin{aligned}\mu_{\mathcal{M}}^{\cup}(\mathrm{d}(S,T))\times\mu_{\mathcal{M}}^{\cup}(\mathrm{d}U) &= \mu_{\mathcal{M}}^{|T|}(\mathrm{d}(S,T))\times\mu_{\mathcal{M}}^{|U|}(\mathrm{d}U)\\ &= \mu_{\mathcal{M}}^{|T|+|U|}(\mathrm{d}(S,T,U))\\ &= \mu_{\mathcal{M}}^{\cup}(\mathrm{d}(S,T,U)).\end{aligned}$$

Comparing with Equation (5), and noting that $|U_i|$ is the number of self-transitions in interval (t_{i-1},t_i), we see both are equal whenever $U_{\mathcal{S}}$ is compatible with $(s_0,S,T,U_{\mathcal{T}})$. The probability density at any incompatible setting of $U_{\mathcal{S}}$ is zero, giving us the desired result. ■

We can now incorporate the likelihoods coming from the observations X. Firstly, note that by assumption, X depends only on the MJP trajectory (s_0,S,T) and not on the auxiliary jumps U. Thus, the conditional distribution of $U_{\mathcal{T}}$ given (s_0,S,T,X) is still the inhomogeneous Poisson process given above. Let $X_{[w_i,w_{i+1})}$ represent the observations in the interval $[w_i,w_{i+1})$ (taking $w_{|W|+1}=t_{end}$). Throughout this interval, the MJP is in state v_i, giving a likelihood term:

$$L_i(v_i) = p(X_{[w_i,w_{i+1})}|\mathbf{S}(t)=v_i \text{ for } t\in[w_i,w_{i+1})). \tag{7}$$

For the case of noisy observations of the MJP state at a discrete set of times T^o, this simplifies to

$$L_i(v_i) = \prod_{j:t_j^o\in[w_i,w_{i+1})} p(X_{t_j^o}|\mathbf{S}(t_j^o)=v_i).$$

Conditioned on the times W, (s_0,V) is a Markov chain with initial distribution π_0, transition matrix B and likelihoods given by Equation (7). We can efficiently resample (s_0,V) using the standard forward filtering-backward sampling (FFBS) algorithm. We provide a description of this algorithm in Appendix A. This cost of such a resampling step is $O(N^2|V|)$, quadratic in the number of states and linear in the length of the chain. Further, any structure in A (e.g., sparsity) is inherited by B and can be exploited easily. Let $(\tilde{s}_0,\tilde{V})$ be the new state sequence. Then $(\tilde{s}_0,\tilde{V},W)$ will correspond to a new MJP path $\tilde{\mathbf{S}}(t)\equiv(\tilde{s}_0,\tilde{S},\tilde{T})$, obtained by discarding virtual jumps from $(\tilde{V},W)$. Effectively, given an MJP path, an iteration of our algorithm corresponds to introducing thinned events, relabelling the thinned and actual transitions using FFBS, and then discarding the new thinned events to obtain a new MJP. We summarize this in Algorithm 2.

Algorithm 2 Blocked Gibbs sampler for an MJP on the interval $[t_{start}, t_{end}]$

Input: A set of observations X, and parameters A (the rate matrix), π_0 (the initial distribution over states) and $\Omega > \max_s(|A_s|)$.
The previous MJP path, $\mathbf{S}(t) \equiv (s_0, S, T)$.
Output: A new MJP trajectory $\tilde{\mathbf{S}}(t) \equiv (\tilde{s}_0, \tilde{S}, \tilde{T})$.

1: Sample $U_{\mathcal{T}} \subset [t_{start}, t_{end}]$ from a Poisson process with piecewise-constant rate

$$R(t) = (\Omega + A_{\mathbf{S}(t)}).$$

Define $W = T \cup U_{\mathcal{T}}$ (in increasing order).

2: Sample a path $(\tilde{s}_0, \tilde{V})$ from a discrete-time Markov chain with $1 + |W|$ steps using the FFBS algorithm. The transition matrix of the Markov chain is $B = \left(I + \frac{A}{\Omega}\right)$ while the initial distribution over states is π_0. The likelihood of state s at step i is

$$L_i(s) = p(X_{[w_i, w_{i+1})} | \mathbf{S}(t) = s \text{ for } t \in [w_i, w_{i+1})).$$

3: Let $\tilde{T}$ be the set of times in W when the Markov chain changes state. Define $\tilde{S}$ as the corresponding set of state values. **Return** $(\tilde{s}_0, \tilde{S}, \tilde{T})$.

Proposition 3 *The auxiliary variable Gibbs sampler described above has the posterior distribution $p(s_0, S, T|X)$ as its stationary distribution. Moreover, if $\Omega > \max_s |A_s|$, the resulting Markov chain is irreducible.*

Proof The first statement follows since the algorithm simply introduces auxiliary variables U, and then conditionally samples V given X and W. For the second, note that if $\Omega > \max_s(|A_s|)$, then the intensity of the subordinating Poisson process is strictly positive. Thus, there is positive probability density of sampling appropriate auxiliary jump times U and moving from any MJP trajectory to any other. ■

Note that it is essential for Ω to be strictly greater than $\max_s |A_s|$; equality is not sufficient for irreducibility. For example, if all diagonal elements of A are equal to Ω, then the Poisson process for $U_{\mathcal{T}}$ will have intensity 0, so that the set of jump times T will never increase.

Since FFBS returns a new state sequence $\tilde{V}$ that is independent of V given W, the only dependence between successive MCMC samples arises because the new candidate jump times include the old jump times, that is, $T \subset W$. This means that the new MJP trajectory has non-zero probability of making a jump at a same time as the old trajectory. Increasing Ω introduces more virtual jumps, and as T becomes a smaller subset of W, we get faster mixing. Of course, increasing Ω makes the HMM chain grow longer, leading to a linear increase in the computational cost per iteration. Thus the parameter Ω allows a trade-off between mixing rate and computational cost. We will study this trade-off in Section 3.5. In all other experiments, we set $\Omega = \max_s(2|A_s|)$ as we find this works quite well, with the samplers typically converging after fewer than 5 iterations.

3.3 Previous Posterior Sampling Schemes

A simple approach when the MJP state is observed at the ends of an interval is rejection sampling: sample paths given the observed start state via Gillespie's algorithm, and reject those that do not end

in the observed end state (Nielsen, 2002). We can extend this to noisy observations by importance sampling or particle filtering (Fan and Shelton, 2008). Recently, Golightly and Wilkinson (2011) have applied particle MCMC methods to correct the bias introduced by standard particle filtering methods. However, these methods are efficient only in situations where the data exerts a relatively weak influence on the unobserved trajectory (compared to the prior): a large state-space or an unlikely end state can result in a large number of rejections or small effective sample sizes (Hobolth and Stone, 2009).

A second approach, more specific to MJPs, integrates out the infinitely many paths of the MJP in between observations using matrix exponentiation (Equation (1)), and uses forward-backward dynamic programming to sum over the states at the finitely many observation times (see Hobolth and Stone, 2009 for a review). Unfortunately, matrix exponentiation is an expensive operation that scales as $O(N^3)$, cubically in the number of states. Moreover, the matrix resulting from matrix exponentiation is dense and any structure (like sparsity), in the rate matrix A cannot be exploited.

A third approach is, like ours, based on the idea of uniformization (Hobolth and Stone, 2009). This proceeds by producing independent posterior samples of the Poisson events W in the interval between observations, and then (like our sampler) running a discrete-time Markov chain on this set of times to sample a new trajectory. However, sampling from the posterior distribution over W is not easy, depending crucially on the observation process, and usually requires a random number of $O(N^3)$ matrix multiplications (as the sampler iterates over the possible number of Poisson events). By contrast, instead of producing independent samples, ours is an MCMC algorithm. At the price of producing dependent samples, our method scales as $O(N^2)$ given a random discretization of time, does not require matrix exponentiations, easily exploits structure in the rate matrix and naturally extends to various extensions of MJPs. Moreover, we demonstrate that our sampler mixes very rapidly. We point out here that as the number of states N increases, if the transition rates $A_{ss'}$, $s \neq s'$, remain $O(1)$, then the uniformization rate Ω and the total number of state transitions are $O(N)$. Thus, our algorithm now scales overall as $O(N^3)$, while the matrix exponentiation-based approach is $O(N^4)$. In either case, whether $A_{ss'}$ is $O(1)$ or $O(1/N)$, our algorithm is an order of magnitude faster.

3.4 Bayesian Inference on the MJP Parameters

In this section we briefly describe how full Bayesian analysis can be performed by placing priors on the MJP parameters A and π_0 and sampling them as part of the MCMC algorithm. Like Fearnhead and Sherlock (2006), we place independent gamma priors on the (negative) diagonal elements of A and independent Dirichlet priors on the transition probabilities. In particular, for all s let $p_{s's} = A_{s's}/|A_s|$ and define the prior:

$$|A_s| \sim \text{Gamma}(\alpha_1, \alpha_2),$$
$$(p_{s's}, s' \neq s) \sim \text{Dirichlet}(\beta).$$

This prior is conjugate, with sufficient statistics for the posterior distribution given a trajectory $\mathbf{S}(t)$ being the total amount of time T_s spent in each state s and the number of transitions $n_{s's}$ from each s to s'. In particular,

$$|A_s|\,|\,(s_0, S, T) \sim \text{Gamma}(\alpha_1 + \textstyle\sum_{s' \neq s} n_{s's}, \alpha_2 + T_s), \quad \text{and} \tag{8}$$
$$(p_{s's}, s' \neq s)|(s_0, S, T) \sim \text{Dirichlet}(\beta + (n_{s's}, s' \neq s)) \tag{9}$$

It is important to note that we resample the rate matrix A conditioned on (s_0, S, T), and *not* (v_0, V, W). A new rate matrix $\tilde{A}$ implies a new uniformization rate $\tilde{\Omega}$, and in the latter case, we must also account for the probability of the Poisson events W under $\tilde{\Omega}$. Besides being more complicated, this coupling between W and Ω can slow down mixing of the MCMC sampler. Thus, we first discard the thinned events U, update A conditioned only on the MJP trajectory, and then reintroduce the thinned events under the new parameters. We can view the sampler of Algorithm 2 as a transition kernel $\mathcal{K}_A((s_0, S, T), (\tilde{s}_0, \tilde{S}, \tilde{T}))$ that preserves the posterior distribution under the rate matrix A. Our overall sampler then alternately updates (s_0, S, T) via the transition kernel $\mathcal{K}_A(\cdot, \cdot)$, and then updates A given (s_0, S, T).

Finally, we can either fix π_0 or (as is sometimes appropriate) set it equal to the stationary distribution of the MJP with rate matrix A. In the latter case, Equations (8) and (9) serve as a Metropolis-Hastings proposal. We accept a proposed $\tilde{A}$ sampled from this distribution with probability equal to the probability of the current initial state under the stationary distribution of $\tilde{A}$. Note that computing this stationary distribution requires solving an $O(N^3)$ eigenvector problem, so that in this case, the overall Gibbs sampler scales cubically even though Algorithm 2 scales quadratically.

3.5 Experiments

We first look at the effect of the parameter Ω on the mixing on the MCMC sampler. We generated a random 5-by-5 matrix A (with hyperparameters $\alpha_1 = \alpha_2 = \beta = 1$), and used this to generate an MJP trajectory with a uniform initial distribution over states. The state of this MJP trajectory was observed via a Poisson process likelihood model (see Section 4), and posterior samples given the observations and A were produced by a C++ implementation of our algorithm. 1000 MCMC runs were performed, each run consisting of 10000 iterations after a burn-in of 1000 iterations. For each run, the number of transitions as well as the time spent in each state was calculated, and effective sample sizes (ESSs) of these statistics (the number of independent samples with the same 'information' as the correlated MCMC samples) were calculated using R-CODA (Plummer et al., 2006). The overall ESS of a run is defined to be the median ESS across all these ESSs.

Figure 3 (left) plots the overall ESS against computation time per run, for different scalings k, where $\Omega = k \max_s |A_s|$. We see that increasing Ω does increase the mixing rate, however the added computational cost quickly swamps out any benefit this might afford. Figure 3 (right) is a similar plot for the case where we also performed Bayesian inference for the MJP parameter A as described in Section 3.4. Now we estimated the ESS of all off-diagonal elements of the matrix A, and the overall ESS of an MCMC run is defined as the median ESS. Interestingly, in this scenario, the ESS is fairly insensitive to Ω, suggesting an 'MCMC within Gibbs' update as proposed here using dependent trajectories is as effective as one using independent trajectories. We found this to be true in general: when embedded within an outer MCMC sampler, our sampler produced similar effective ESSs as an MJP sampler that produces independent trajectories. The latter is typically more expensive, and in any case, we will show that the computational savings provided by our sampler far outweigh the cost of dependent trajectories.

In light of Figure 3, for all subsequent experiments we set $\Omega = 2 \max_s |A_s|$. Figure 4 shows the initial burn-in of a sampler with this setting for different initializations. The vertical axis shows the number of state transitions in the MJP trajectory of each iteration. This quantity quickly reaches its equilibrium value within a few iterations.

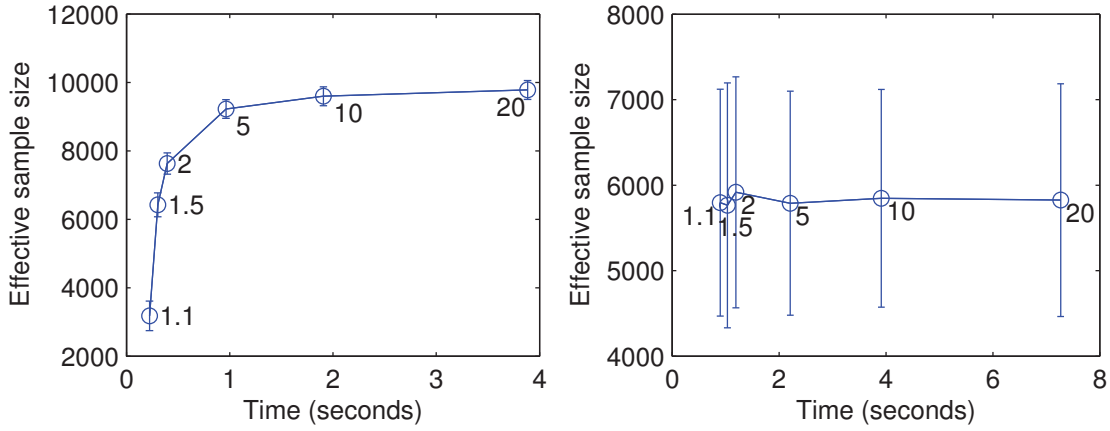


Figure 3: Effective sample sizes vs computation times for different scalings of Ω for (left) a fixed rate matrix A and (right) Bayesian inference on A. Whiskers are quartiles over 1000 runs.

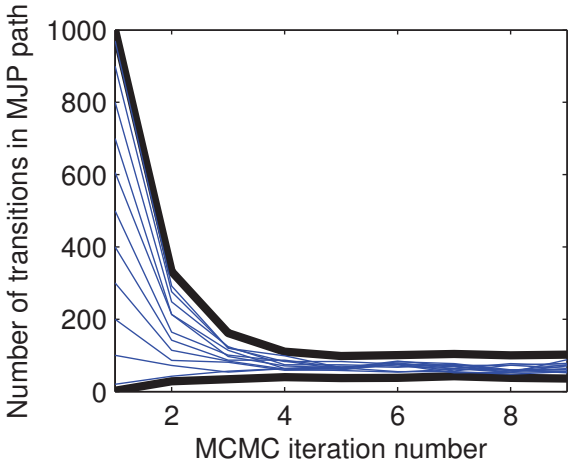


Figure 4: Trace plot of the number of MJP transitions for different initializations. Black lines are the maximum and minimum number of MJP transitions for each iteration, over all initializations.

4. Markov-Modulated Poisson Processes

A Markov modulated Poisson process (MMPP) is a doubly-stochastic Poisson process whose intensity function is piecewise-constant and distributed according to a Markov jump process. Suppose the MJP $(\mathbf{S}(t), t \in [t_{start}, t_{end}])$ has N states, and is parametrized by an initial distribution over states π_0 and a rate matrix A. Associate with each state s a nonnegative constant λ_s called the emission rate of state s. Let O be a set of points drawn from a Poisson process with piecewise-constant rate $R(t) = \lambda_{\mathbf{S}(t)}$. Note that O is unrelated to the subordinating Poisson process from the uniformization-based construction of the MJP, and we call it the output Poisson process. The Poisson observations

O effectively form a continuous-time observation of the latent MJP, with the *absence* of Poisson events also informative about the MJP state. MMPPs have been used to model phenomenon like the distribution of rare DNA motifs along a gene (Fearnhead and Sherlock, 2006), photon arrival in single molecule fluorescence experiments (Burzykowski et al., 2003), and requests to web servers (Scott and Smyth, 2003).

Fearnhead and Sherlock (2006) developed an exact sampler for MMPPs based on a dynamic program for calculating the probability of O marginalizing out the MJP trajectory. The dynamic program keeps track of the probability of the MMPP emitting all Poisson events prior to a time t and ending in MJP state s. The dynamic program then proceeds by iterating over all Poisson events in O in increasing order, at each iteration updating probabilities using matrix exponentiation. A backward sampling step then draws an exact posterior sample of the MJP trajectory $(\mathbf{S}(t), t \in O)$ evaluated at the times in O. Finally a uniformization-based endpoint conditioned MJP sampler is used to fill in the MJP trajectory between every pair of times in O.

The main advantage of this method is that it produces independent posterior samples. It does this at the price of being fairly complicated and computationally intensive. Moreover, it has the disadvantage of operating at the time scale of the Poisson observations rather than the dynamics of the latent MJP. For high Poisson rates, the number of matrix exponentiations will be high even if the underlying MJP has very low transition rates. This can lead to an inefficient algorithm.

Our MCMC sampler outlined in the previous section can be straightforwardly extended to the MMPP without any of these disadvantages. Resampling the auxiliary jump events (step 1 in algorithm 2) remains unaffected, since conditioned on the current MJP trajectory, they are independent of the observations O. Step 2 requires calculating the emission likelihoods $L_i(s)$, which is simply given by:

$$L_i(s) = (\lambda_s)^{|O_i|} \exp\left(-\lambda_s (w_{i+1} - w_i)\right),$$

$|O_i|$ being the number of events of O in the interval $[w_i, w_{i+1})$. Note that evaluating this likelihood only requires counting the number of observed Poisson events between every successive pair of times in W. Compared to our algorithm, the approach of Fearnhead and Sherlock (2006) is much more involved and inefficient.

4.1 Experiments

In the following, we compare a C++ implementation of our algorithm with an implementation[2] of the algorithm of Fearnhead and Sherlock (2006), coded in C. We performed fully Bayesian inference, sampling both the MJP parameters (as described in Section 3.4) and the Poisson rates λ_s (conjugate gamma priors were placed on these). In all instances, our algorithm did significantly better, the performance improvement increasing with the complexity of the problem.

In the first set of experiments, the dimension of the latent MJP was fixed to 5. The prior on the rate matrix A had parameters $\alpha_1 = \alpha_2 = \beta = 1$ (see Section 3.4). The shape parameter of the gamma prior on the emission rate of state s, λ_s, was set to s (thereby breaking symmetry across states); the scale parameter was fixed at 1. 10 draws of O were simulated using the MMPP. For each observed O, both MCMC algorithms were run for 1000 burn-in iterations followed by 10000 iterations where samples were collected. For each run, the ESS for each parameter was estimated using R-CODA, and the overall ESS was defined to be the median ESS over all parameters.

2. Code was downloaded from Chris Sherlock's webpage.

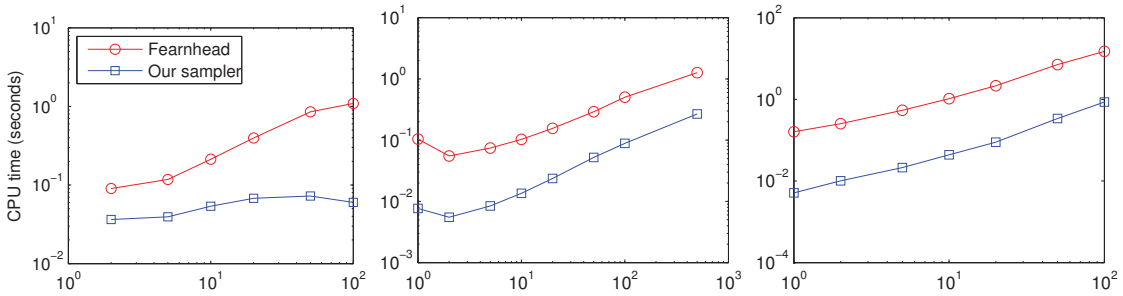


Figure 5: CPU time to produce 100 effective samples as we observe (left) increasing number of Poisson events in an interval of length 10, (centre) 10 Poisson events over increasing time intervals, and (right) increasing intervals with the number of events increasing on average.

Figure 5 reports the average computation times required by each algorithm to produce 100 effective samples, under different scenarios. The leftmost plot shows the computation times as a function of the numbers of Poisson events observed in an interval of fixed length 10. For our sampler, increasing the number of observed events leaves the computation time largely unaffected, while for the sampler of Fearnhead and Sherlock (2006), this increases quite significantly. This reiterates the point that our sampler works at the time scale of the latent MJP, while Fearnhead and Sherlock (2006) work at the time scale of the observed Poisson process. In the middle plot, we fix the number of observed Poisson events to 10, while increasing the length of the observation interval instead, while in the rightmost plot, we increase both the interval length and the average number of observations in that interval. In both cases, our sampler again offers increased efficiency of up to two orders of magnitude. In fact, the only problems where we observed the sampler of Fearnhead and Sherlock (2006) to outperform ours were low-dimensional problems with only a few Poisson observations in a long interval, and with one very unstable state. A few very stable MJP states and a few very unstable ones results in a high uniformization rate Ω but only a few state transitions. The resulting large number of virtual jumps can make our sampler inefficient.

In Figure 6, we plot the time to produce 100 effective samples as the number of states of the latent MJP increases. Here, we fixed the number of Poisson observations to 10 over an interval of length 10. We see that our sampler (plotted with squares) offers substantial speed-up over the sampler of Fearnhead and Sherlock (2006) (plotted with circles). We see that for both samplers computation time scales cubically with the latent dimension. However, recall that this cubic scaling is not a property of our MJP trajectory sampler; rather it is a consequence of using the equilibrium distribution of a sampled rate matrix as the initial distribution over states, which requires calculating an eigenvector of a proposed rate matrix. If we fix the initial distribution over states (to the discrete uniform distribution), giving the line plotted with inverted triangles in the figure, we observe that our sampler scales quadratically.

5. Continuous-Time Bayesian Networks (CTBNs)

Continuous-time Bayesian networks (CTBNs) are compact, multi-component representations of MJPs with structured rate matrices (Nodelman et al., 2002). Special instances of these models

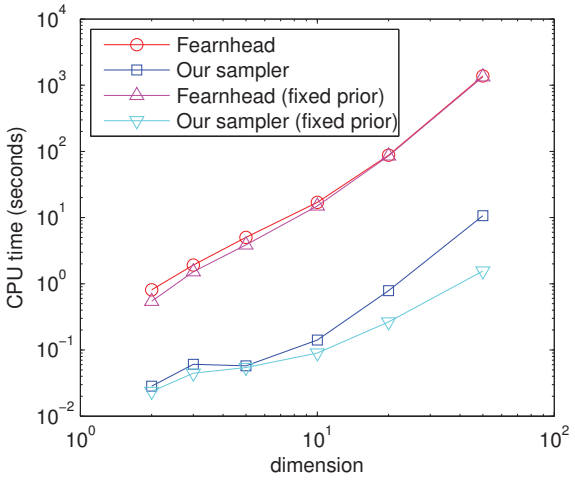


Figure 6: CPU time to produce 100 effective samples as the MJP dimension increases

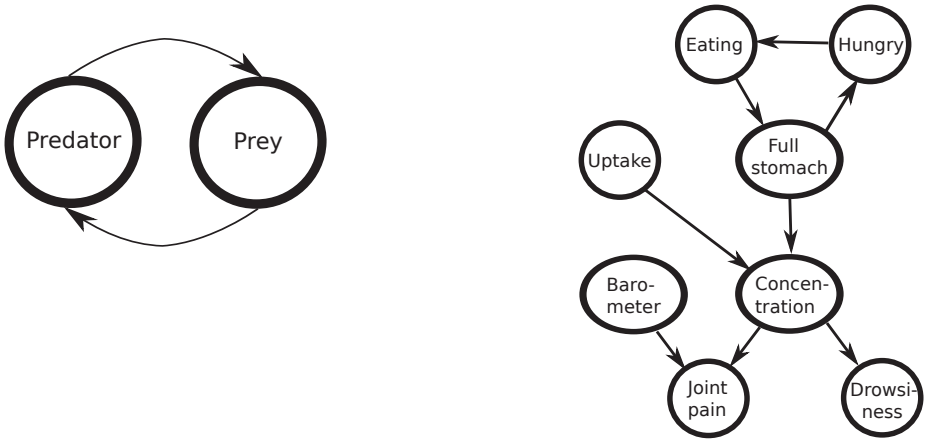


Figure 7: The predator-prey network (left) and the drug-effect CTBN (right)

have long existed in the literature, particularly stochastic kinetic models like the Lotka-Volterra equations, which describe interacting populations of animal species, chemical reactants or gene regulatory networks (Wilkinson, 2009). There have also been a number of related developments, see for example Bolch et al. (1998) or Didelez (2008). For concreteness however, we shall focus on CTBNs, a formalism introduced in Nodelman et al. (2002) to harness the representational power of Bayesian networks to characterize structured MJPs.

Just as the familiar Bayesian network uses a product of conditional probability tables to represent a much larger probability table, so too a CTBN represents a structured rate matrix with smaller conditional rate matrices. An m-component CTBN represents the state of an MJP at time t with the states of m nodes $\mathbf{S}^1(t), \ldots, \mathbf{S}^m(t)$ in a directed (and possibly cyclic) graph $\mathcal{G}$. Figure 7 shows two CTBNs, the 'predator-prey network' and the 'drug-effect network'. The former is a CTBN governed by the Lotka-Volterra equations (see subsection 5.3.1), while the latter is used to model the dependencies in events leading to and following a patient taking a drug (Nodelman et al., 2002).

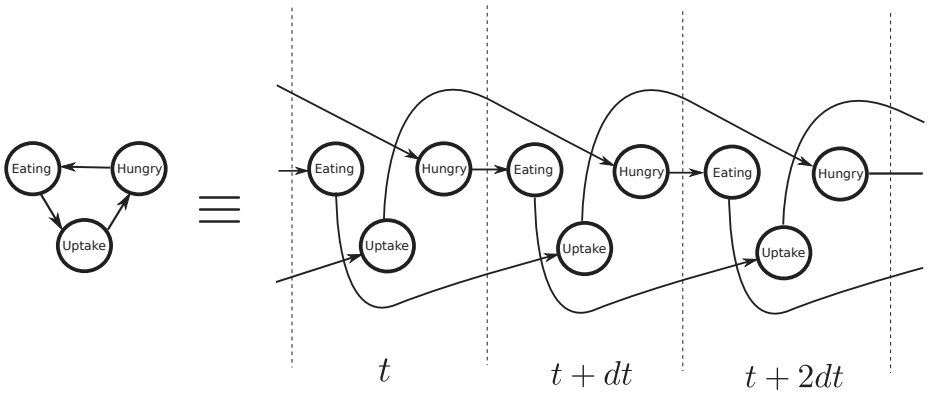


Figure 8: Expanded CTBN

Intuitively, each node of the CTBN acts as an MJP with an instantaneous rate matrix that depends on the current configuration of its parents (and not its children, although the presence of directed cycles means a child can be a parent as well). The trajectories of all nodes are piecewise constant, and when a node changes state, the event rates of all its children change. The graph $\mathcal{G}$ and the set of rate matrices (one for each node and for each configuration of its parents) characterize the dynamics of the CTBN, the former describing the structure of the dependencies between various components, and the latter quantifying this. Completing the specification of the CTBN is an initial distribution π_0 over the state of nodes, possibly specified via a Bayesian network.

It is convenient to think of a CTBN as a compact representation of an expanded (and now acyclic) graph, consisting of the nodes of $\mathcal{G}$ repeated infinitely along a continuum (viz. time). In this graph, arrows lead from a node at a time t to instances of its children at time $t + \mathrm{d}t$. Figure 8 displays this for a section of the drug-effect CTBN. The rates associated with a particular node at time $t + \mathrm{d}t$ are determined by the configuration of its parents at time t. Figure 8 is the continuous-time limit of a class of discrete-time models called dynamic Bayesian networks (DBNs) (Murphy, 2002). In a DBN, the state of a node at stage $i+1$ is dependent upon the configuration of its parents at stage i. Just as MJPs are continuous-time limits of discrete-time Markov chains, CTBNs are also continuous-time limits of DBNs.

It is possible to combine all local rate matrices of a CTBN into one global rate matrix (see Nodelman et al., 2002), resulting in a simple MJP whose state-space is the product state-space of all component nodes. Consequently, it possible, conceptually at least, to directly sample a trajectory over an interval $[t_{start}, t_{end}]$ using Gillespie's algorithm. However, with an eye towards inference, Algorithm 3 describes a generative process that exploits the structure in the graph $\mathcal{G}$. Like Section 2, we represent the trajectory of the CTBN, $\mathbf{S}(t)$, with the initial state s_0 and the pair of sequences (S,T). Let the CTBN have m nodes. Now, s_i, the ith element of S, is an m-component vector representing the states of all nodes at t_i, the time of the ith state change of the CTBN. We write this as $s_i = (s_i^1, \cdots, s_i^m)$. Let k_i identify the component of the CTBN that changed state at t_i. The rate matrix of a node n varies over time as the configuration of its parents changes, and we will write $A^{n,t}$ for the relevant matrix at time t. Following Equation (2), we can write down the probability density

of (s_0, S, T) as

$$p(s_0, S, T) = \pi_0(s_0) \left(\prod_{i=1}^{|T|} A^{k_i, t_{i-1}}_{s_i^{k_i} s_{i-1}^{k_i}} \right) \exp\left(- \sum_{k=1}^{m} \int_{t_{start}}^{t_{end}} |A^{k,t}_{\mathbf{S}^k(t)}| \mathrm{d}t \right). \tag{10}$$

Algorithm 3 Algorithm to sample a CTBN trajectory on the interval $[t_{start}, t_{end}]$

Input: The CTBN graph $\mathcal{G}$, a set of rate matrices $\{A\}$ for all nodes and for all parent configurations and an initial distribution over states π_0.

Output: A CTBN trajectory $\mathbf{S}(t) \equiv (s_0, S, T)$.

1: Draw an initial configuration $s_0 \equiv (s_0^1, s_0^2, \ldots) \sim \pi_0$. Set $t_0 = t_{start}$ and $i = 0$.
2: **loop**
3: For each node k, draw $z^k \sim \exp(|A^{k,t_i}_{s_i^k}|)$.
4: Let $k_{i+1} = \mathrm{argmin}_k z^k$ be the first node to change state.
5: **If** $t_i + z^{k_{i+1}} > t_{end}$ **then return** $(s_0, \ldots, s_i, t_1, \ldots, t_i)$ and **stop**.
6: Increment i and let $t_i = t_{i-1} + z^{k_i}$ be the next jump time.
7: Let $s' = s_{i-1}^{k_i}$ be the previous state of node k_i.
8: Set $s_i^{k_i} = s$ with probability proportional to $A^{k_i, t_{i-1}}_{ss'}$ for each $s \neq s'$.
9: Set $s_i^k = s_{i-1}^k$ for all $k \neq k_i$.
10: **end loop**

5.1 Inference in CTBNs

We now consider the problem of posterior inference over trajectories given some observations. Write the parents and children of a node k as $\mathcal{P}(k)$ and $\mathcal{C}(k)$ respectively. Let $\mathcal{MB}(k)$ be the Markov blanket of node k, which consists of its parents, children, and the parents of its children. Given the entire trajectories of all nodes in $\mathcal{MB}(k)$, node k is independent of all other nodes in the network (Nodelman et al., 2002) (see also Equation (12) below). This suggests a Gibbs sampling scheme where the trajectory of each node is resampled given the configuration of its Markov blanket. This approach was followed by El-Hay et al. (2008).

However, even without any associated observations, sampling a node trajectory conditioned on the complete trajectory of its Markov blanket is not straightforward. To see this, rearrange the terms of Equation (10) to give

$$p(s_0, S, T) = \pi_0(s_0) \prod_{k=1}^{m} \phi(S^k, T^k | s_0, S^{\mathcal{P}(k)}, T^{\mathcal{P}(k)}), \quad \text{and}$$

$$\phi(S^k, T^k | s_0, S^{\mathcal{P}(k)}, T^{\mathcal{P}(k)}) = \left(\prod_{i: k_i = k} A^{k, t_{i-1}}_{s_i^k s_{i-1}^k} \right) \exp\left(- \int_{t_{start}}^{t_{end}} |A^{k,t}_{\mathbf{S}^k(t)}| \mathrm{d}t \right), \tag{11}$$

where for any set of nodes B, (s_0^B, S^B, T^B) represents the associated trajectories. Note that the $\phi(\cdot)$ terms are not conditional densities given parent trajectories, since the graph $\mathcal{G}$ can be cyclic. We must also account for the trajectories of node k's children, so that the conditional density of

(s_0^k, S^k, T^k) is actually

$$p(s_0^k, S^k, T^k | s_0^{\neg k}, S^{\neg k}, T^{\neg k}) \propto \pi_0(s_0^k | s_0^{\neg k}) \phi(S^k, T^k | s_0, S^{\mathcal{P}(k)}, T^{\mathcal{P}(k)}) \cdot \prod_{c \in \mathcal{C}(k)} \phi(S^c, T^c | s_0, S^{\mathcal{P}(c)}, T^{\mathcal{P}(c)}). \quad (12)$$

Here $\neg k$ denotes all nodes other than k. Thus, even over an interval of time where the parent configuration remains constant, the conditional distribution of the trajectory of a node is not a homogeneous MJP because of the effect of the node's children, which act as 'observations' that are continuously observed. For any child c, if $A^{c,t}$ is constant over t, the corresponding $\phi(\cdot)$ is the density of an MJP given the initial state. Since $A^{c,t}$ varies in a piecewise-constant manner according to the state of k, the $\phi(\cdot)$ term is actually the density of a piecewise-inhomogeneous MJP. Effectively, we have a 'MJP-modulated MJP', so that the inference problem here is a generalization of that for the MMPP of Section 4.

El-Hay et al. (2008) described a matrix-exponentiation-based algorithm to update the trajectory of node k. At a high-level their algorithm is similar to Fearnhead and Sherlock (2006) for MMPPs, with the Poisson observations of the MMPP generalized to transitions in the trajectories of child nodes. Consequently, it uses an expensive forward-backward algorithm involving matrix exponentiations. In addition, El-Hay et al. (2008) resort to discretizing time via a binary search to obtain the transition times upto machine accuracy.

5.2 Auxiliary Variable Gibbs Sampling for CTBNs

We now show how our uniformization-based sampler can easily be adapted to conditionally sample a trajectory for node k without resorting to approximations. In the following, for notational simplicity. we will drop the superscript k whenever it is clear from context. For node k, the MJP trajectory (s_0, S, T) has a uniformized construction from a subordinating Poisson process. The piecewise constant trajectories of the parents of k imply that the MJP is piecewise homogeneous, and we will use a piecewise constant rate Ω^t which dominates the associated transition rates, that is, $\Omega^t > |A_s^{k,t}|$ for all s. This allows the dominating rate to 'adapt' to the local transition rates, and is more efficient when, for example, the transition rates associated with different parent configurations are markedly different. Recall also that our algorithm first reconstructs the thinned Poisson events $U_\mathcal{T}$ using a piecewise homogeneous Poisson process with rate $(\Omega^t + A^{k,t}_{\mathbf{S}(t)})$, and then updates the trajectory using the forward-backward algorithm (so that $W = T \cup U_\mathcal{T}$ forms the candidate transitions times of the MJP).

In the present CTBN context, just as the subordinating Poisson process is inhomogeneous, so too the Markov chain used for the forward-backward algorithm will have different transition matrices at different times. In particular, the transition matrix at a time w_i (where $W = (w_1, \ldots, w_{|W|})$) is

$$B_i = I + \frac{A^{k,w_i}}{\Omega^{w_i}}.$$

Finally, we need also to specify the likelihood function $L_i(s)$ accounting for the trajectories of the children in addition to actual observations in each time interval $[w_i, w_{i+1})$. From Equations (11) and (12), this is given by

$$L_i(s) = L_i^O(s) \prod_{c \in \mathcal{C}(k)} \left(\prod_{j: k_j = k, t_j \in [w_i, w_{i+1})} A^{k,t_{j-1}}_{s_j^k s_{j-1}^k} \right) \exp\left(- \int_{w_i}^{w_{i+1}} |A^{k,t}_{\mathbf{S}^k(t)}| \mathrm{d}t \right),$$

where $L_i^O(s)$ is the likelihood coming from actual observations dependent on the state of node k in the time interval. Note that the likelihood above depends only on the number of transitions each of the children make as well as how much time they spend in each state, for each parent configuration.

The new trajectory $\tilde{\mathbf{S}}^k(t)$ is now obtained using the forward-filtering backward-sampling algorithm, with the given inhomogeneous transition matrices and likelihood functions. The following proposition now follows directly from our previous results in Section 3:

Proposition 4 *The auxiliary variable Gibbs sampler described above converges to the posterior distribution over the CTBN sample paths.*

Note that our algorithm produces a new trajectory that is dependent, through T, on the previous trajectory (unlike a true Gibbs update as in El-Hay et al. (2008) where they are independent). However, we find that since the update is part of an overall Gibbs cycle over nodes of the CTBN, the mixing rate is actually dominated by dependence across nodes. Thus, a true Gibbs update has negligible benefit towards mixing, while being more expensive computationally.

5.3 Experiments

In the following, we evaluate a C++ implementation of our algorithm on a number of CTBNs. As before, the dominating rate Ω^t was set to $\max_s 2|A_s^{k,t}|$.

5.3.1 THE LOTKA-VOLTERRA PROCESS

We first apply our sampler to the Lotka-Volterra process (Wilkinson, 2009; Opper and Sanguinetti, 2007). Commonly referred to as the predator-prey model, this describes the evolution of two interacting populations of 'prey' and 'predator' species. The two species form the two nodes of a cyclic CTBN (Figure 7 (left)), whose states x and y represent the sizes of the prey and predator populations. The process rates are given by

$$A(\{x,y\} \to \{x+1,y\}) = \alpha x, \qquad A(\{x,y\} \to \{x-1,y\}) = \beta xy,$$
$$A(\{x,y\} \to \{x,y+1\}) = \delta xy, \qquad A(\{x,y\} \to \{x,y-1\}) = \gamma y,$$

where we set the parameters as follows: $\alpha = 5\times10^{-4}, \beta = 1\times10^{-4}, \gamma = 5\times10^{-4}, \delta = 1\times10^{-4}$. All other rates are 0. This defines two infinite sets of infinite-dimensional conditional rate matrices. In its present form, our sampler cannot handle this infinite state-space (but see Rao and Teh, 2012). Like Opper and Sanguinetti (2007), we limit the maximum number of individuals of each species to 200, leaving us with 400 rate matrices of size 200×200. Note that these matrices are tridiagonal and very sparse: at any time the size of each population can change by at most one. Consequently, the complexity of our algorithm scales *linearly* with the number of states (we did not modify our code to exploit this structure, though this is straightforward). A 'true' path of predator-prey population sizes was sampled from this process, and its state at time $t = 0$ was observed noiselessly. Additionally 15 noisy observations were generated, spaced uniformly at intervals of 100. The noise process was:

$$p(X(t)|\mathbf{S}(t)) \propto \frac{1}{2^{|X(t)-\mathbf{S}(t)|}+10^{-6}}.$$

Given these observations (as well as the true parameter values), we approximated the posterior distribution over paths by two methods: using 1000 samples from our MCMC sampler (with a

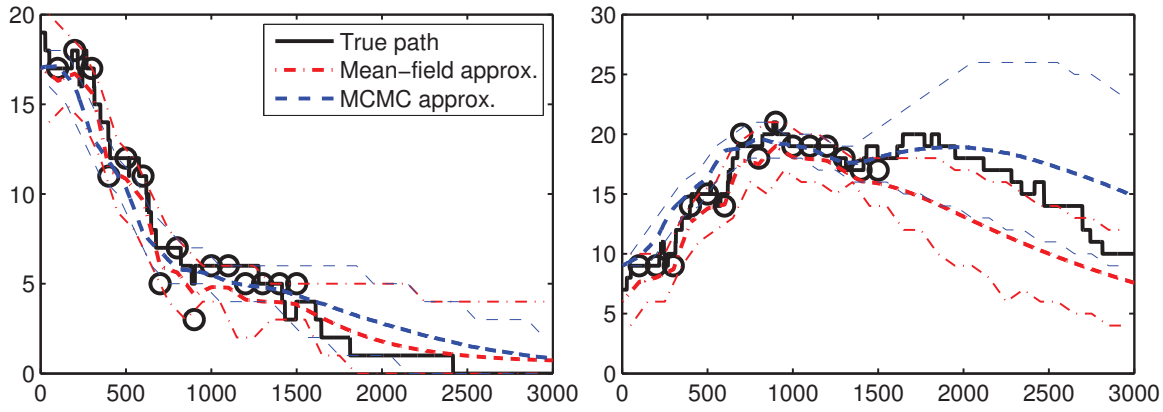


Figure 9: Posterior (mean and 90% credible intervals) over (left) prey and (right) predator paths (observations (circles) were available only until 1500).

burn-in period of 100) and using the mean-field (MF) approximation of Opper and Sanguinetti (2007).[3] We could not apply the implementation of the Gibbs sampler of El-Hay et al. (2008) (see Section 5.4) to a state-space and time-interval this large. Figure 9 shows the true paths (in black), the observations (as circles) as well as the posterior means and 90% credible intervals produced by the two algorithms for the prey (left) and predator (right) populations. As can be seen, both algorithms do well over the first half of the interval where data is present. In the second half, the MF algorithm appears to underestimate the predicted size of the predator population. On the other hand, the MCMC posterior reflects the true trajectory better. In general, we found the MF algorithm to underestimate the posterior variance in the MJP trajectories, especially over regions with few observations.

5.4 Average Relative Error vs Number of Samples

For the remaining experiments, we compared our sampler with the Gibbs sampler of El-Hay et al. (2008). For this comparison, we used the CTBN-RLE package of Shelton et al. (2010) (also implemented in C++). In all our experiments, as with the MMPP, we found our algorithm to be significantly faster, especially for larger problems. To prevent details of the two implementations from clouding the picture and to reiterate the benefit afforded by avoiding matrix exponentiations, we also measured the amount of time CTBN-RLE spent exponentiating matrices. This constituted between 10% to 70% of the total running time of their algorithm. In the plots we refer to this as 'El Hay et al. (Matrix Exp.)'. We found that our algorithm took less time than even this.

In our first experiment, we followed El-Hay et al. (2008) in studying how average relative error varies with the number of samples from the Markov chain. Average relative error is defined by $\sum_j \frac{|\hat{\theta}_j - \theta_j|}{\theta_j}$, and measures the total normalized difference between empirical ($\hat{\theta}_j$) and true (θ_j) averages of sufficient statistics of the posterior. The statistics in question are the time spent by each node in different states as well as the number of transitions from each state to the others. The exact

3. We thank Guido Sanguinetti for providing us with his code.

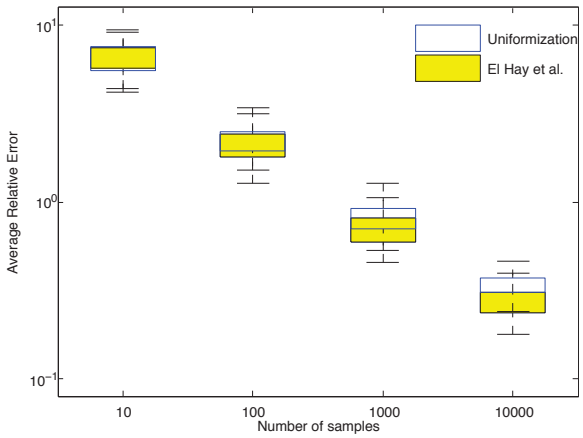


Figure 10: Average relative error vs number of samples for 1000 independent runs; burn-in = 200. Note that in this scenario, uniformization was about 12 times faster, so that for the same computational effort, it produces significantly lower errors.

values were calculated by numerical integration when possible, otherwise from a very long run of CTBN-RLE.

As in El-Hay et al. (2008), we consider a CTBN with the topology of a chain, consisting of 5 nodes, each with 5 states. The states of the nodes were observed at times 0 and 20 and we produced endpoint-conditioned posterior samples of paths over the time interval $[0, 20]$. We calculate the average relative error as a function of the number of samples, with a burn-in of 200 samples. Figure 10 shows the results from running 1000 independent chains for both samplers. Not surprisingly, the sampler of El-Hay et al. (2008), which produces conditionally independent samples, has slightly lower errors. However the difference in relative errors is minor, and is negligible when considering the dramatic (sometimes up to two orders of magnitude; see below) speed improvements of our algorithm. For instance, to produce the 10000 samples, the El-Hay et al. (2008) sampler took about 6 minutes, while our sampler ran in about 30 seconds.

5.5 Time Requirements for the Chain-Shaped CTBN

In the next two experiments, we compare the times required by CTBN-RLE and our uniformization-based sampler to produce 100 effective samples as the size of the chain-shaped CTBN increased in different ways. In the first cases, we increased the length of the chain, and in the second, the dimensionality of each node. In both cases, we produced posterior samples from an endpoint-conditioned CTBN with random gamma distributed parameters.

The time requirements were estimated from runs of 10000 samples after a burn-in period of 1000 iterations. Since CTBN-RLE does not support Bayesian inference for CTBN parameters, we kept these fixed to the truth. To produce ESS estimates, we counted the number of transitions of each node and the amount of time spent in each state, and for each MCMC run, we estimated the ESS of these quantities. Like in Section 4.1, the overall ESS is the median of these estimates. Each point in the figures is an average over 10 simulations.

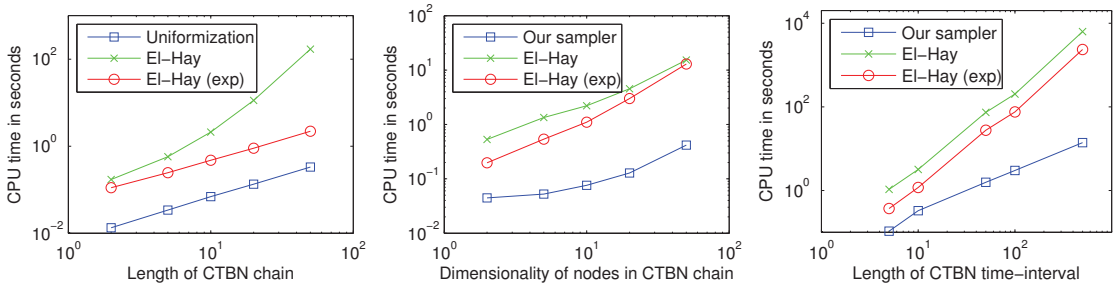


Figure 11: CPU time vs (left) length of CTBN chain (centre) number of states of CTBN nodes (right) time interval of CTBN paths.

In the first of these experiments, we measured the times to produce 100 effective samples for the chain-shaped CTBN described above, as the number of nodes in the chain (i.e., its length) increases. The leftmost plot in Figure 11 shows the results. As might be expected, the time required by our algorithm grows linearly with the number of nodes. For El-Hay et al. (2008), the cost of the algorithm grows faster than linear, and quickly becoming unmanageable. The time spent calculating matrix exponentials *does* grow linearly, however our uniformization-based sampler always takes less time than even this.

Next, we kept the length of the chain fixed at 5, instead increasing the number of states per node. As seen in the middle plot, once again, our sampler is always faster. Asymptotically, we expect our sampler to scale as $O(N^2)$ and El-Hay et al. (2008) as $O(N^3)$. While we have not hit that regime yet, we can see that the cost of our sampler grows more slowly with the number of states.

5.6 Time Requirements for the Drug-Effect CTBN

Our final experiment, reported in the rightmost plot of Figure 11, measures the time required as the interval length $(t_{end} - t_{start})$ increases. For this experiment, we used the drug-effect network shown in Figure 7, where the parameters were set to standard values (obtained from CTBN-RLE) and the state of the network was fully observed at the beginning and end times. Again, our algorithm is the faster of the two, showing a linear increases in computational costs with the length of the interval. It is worth pointing out here that the algorithm of El-Hay et al. (2008) has a 'precision' parameter, and that by reducing the desired temporal precision, faster performance can be obtained. However, since our sampler produces *exact* samples (up to numerical precision), our comparison is fair. In the above experiments, we left this parameter at its default value.

6. Discussion

We proposed a novel Markov chain Monte Carlo sampling method for Markov jump processes. Our method exploits the simplification of the structure of the MJP resulting from the introduction of auxiliary variables via the idea of uniformization. This constructs a Markov jump process by subordinating a Markov chain to a Poisson process, and amounts to running a Markov chain on a random discretization of time. Our sampler is a blocked Gibbs sampler in this augmented represen-

tation and proceeds by alternately resampling the discretization given the Markov chain and vice versa. Experimentally, we find that this auxiliary variable Gibbs sampler is computationally very efficient. The sampler easily generalizes to other MJP-based models, and we presented samplers for Markov-modulated Poisson processes and continuous-time Bayesian networks. In our experiments, we showed significant speed-up compared to state-of-the-art samplers for both.

Our method opens a number of avenues worth exploring. One concerns the subordinating Poisson rate Ω which acts as a free-parameter of the sampler. While our heuristic of setting this to $\max_s 2|A_s|$ worked well in our experiments, this may not be the case for rate matrices with widely varying transition rates. A possible approach is to 'learn' a good setting of this parameter via adaptive MCMC methods. More fundamentally, it would be interesting to investigate if theoretical claims can be made about the 'best' setting of this parameter under some measures of mixing speed and computational cost.

Next, there are a number of immediate generalizations of our sampler. First, our algorithm is easily applicable to inhomogeneous Markov jump processes where techniques based on matrix exponentiation cannot be applied. Following recent work (Rao and Teh, 2011b), we can also look at generalizing our sampler to semi-Markov processes where the holding times of the states follow non-exponential distributions. These models find applications in fields like biostatistics, neuroscience and queueing theory (Mode and Pickens, 1988). By combining our technique with slice sampling ideas (Neal, 2003), we can explore Markov jump processes with countably infinite state spaces. Another generalization concerns MJPs with unbounded rate matrices. For the predator-prey model, we avoided this problem by bounding the maximum population sizes; otherwise it is impossible to choose a dominating Ω. Of course, in practical settings, any trajectory from this process is bounded with probability 1, and we can extend our method to this case by treating Ω as a trajectory dependent random variable. For some work in this direction, we refer to Rao and Teh (2012).

Acknowledgments

This work was done while both authors were at the Gatsby Computational Neuroscience Unit, University College London. We would like to acknowledge the Gatsby Charitable Foundation for generous funding. We thank the associate editor and the anonymous reviewers for useful comments.

Appendix A. The Forward-Filtering Backward-Sampling (FFBS) Algorithm

For completeness, we include a description of the forward-filtering backward-sampling algorithm for discrete-time Markov chains. The earliest references for this that we are aware of are Früwirth-Schnatter (1994) and Carter and Kohn (1996).

Let S_t, $t \in \{0, \cdots T\}$ be a discrete-time Markov chain with a discrete state space $\mathcal{S} \equiv \{1, \cdots N\}$. We allow the chain to be inhomogeneous, with B^t being the state transition matrix at time t (so that $p(S_{t+1} = s' | S_t = s) = B^t_{s's}$). Let π_0 be the initial distribution over states at $t = 0$. Let O^t be a noisy observation of the state at time t, with the likelihood given by $L^t(s) = p(O^t | S^t = s)$. Given a set of observations $O = (O^0, \cdots, O^T)$, FFBS returns an independent posterior sample of the state vector.

Define $\alpha^t(s) = p(O^0, \cdots, O^{t-1}, S^t = s)$. From the Markov property, we have the following recursion:

$$\alpha^{t+1}(s') = \sum_{s=1}^{N} \alpha^t(s) L^t(s) B^t_{s's}.$$

Calculating this for all N values of s' takes $O(N^2)$ computation, and a forward pass through all T times is $O(TN^2)$. At the end of the forward pass, we have a vector

$$\beta^T(s) := L^T(s)\alpha^T(s) = p(O, S_T = s) \propto p(S_T = s|O).$$

It is easy to sample a realization of S_T from this. Next, note that

$$\begin{aligned} p(S_t = s|S_{t+1} = s', O) &\propto p(S_t = s, S_{t+1} = s', O) \\ &= \alpha^t(s) B^t_{s's} L^t(s) p(O^{t+1}, \cdots, O^T|S_{t+1} = s') \\ &\propto \alpha^t(s) B^t_{s's} L^t(s), \end{aligned}$$

where the second equality follows from the Markov property. This too is easy to sample from, and the backward pass of FFBS successively samples S_{T-1} to S_0. We thus have a sample $(S_0, \cdots, S_T)$. The overall algorithm is given below:

Algorithm 4 The forward-filtering backward-sampling algorithm

Input: An initial distribution over states π_0, a sequence of transition matrices B^t, a sequence of observations $O = (O_1, \cdots O_T)$ with likelihoods $L^t(s) = p(O^t|S^t = s)$.

Output: A realization of the Markov chain $(S_0, \cdots, S_T)$.

Set $\alpha^0(s) = \pi_0(s)$.
for $t = 1 \to T$ **do**
    $\alpha^t(s') = \sum_{s=1}^{N} \left(\alpha^{t-1}(s) L^{t-1}(s) B^{t-1}_{s's}\right)$ for $s' \in \{1, \cdots, N\}$.
end for
Sample $S_T \sim \beta^T(\cdot)$, where $\beta^T(s) := L^T(s)\alpha^T(s)$.
for $t = T \to 0$ **do**
    Define $\beta^t(s) = \alpha^t(s) B^t_{S_{t+1}s} L^t(s)$.
    Sample $S_t \sim \beta^t(\cdot)$.
end for
return $(S_0, \cdots, S_T)$.

References

G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. Wiley-Interscience, New York, NY, USA, 1998. ISBN 0-471-19366-6.

R. J. Boys, D. J. Wilkinson, and T. B. L. Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, 18(2):125–135, 2008.

L. Breuer. *From Markov Jump Processes to Spatial Queues*. Springer, 2003. ISBN 978-1-4020-1104-7.

T. Burzykowski, J. Szubiakowski, and T. Rydén. Analysis of photon count data from single-molecule fluorescence experiments. *Chemical Physics*, 288(2-3):291–307, 2003.

C. K. Carter and R. Kohn. Markov chain Monte Carlo in conditionally Gaussian state space models. *Biometrika*, 83:589–601, 1996.

E. Çinlar. *Introduction to Stochastic Processes*. Prentice Hall, 1975.

I. Cohn, T. El-Hay, N. Friedman, and R. Kupferman. Mean field variational approximation for continuous-time Bayesian networks. *J. Mach. Learn. Res.*, 11:2745–2783, December 2010. ISSN 1532-4435.

D. J. Daley and D. Vere-Jones. *An Introduction to the Theory of Point Processes*. Springer, 2008.

V. Didelez. Graphical models for marked point processes based on local independence. *Journal of the Royal Statistical Society: Series B*, 70(1):245–264, 2008. ISSN 1467-9868.

T. El-Hay, N. Friedman, and R. Kupferman. Gibbs sampling in factorized continuous-time Markov processes. In *Proceedings of the Twenty-fourth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 169–178, 2008.

R. Elliott and C. Osakwe. Option pricing for pure jump processes with Markov switching compensators. *Finance and Stochastics*, 10:250–275, 2006.

Y. Fan and C. R. Shelton. Sampling for approximate inference in continuous time Bayesian networks. In *Tenth International Symposium on Artificial Intelligence and Mathematics*, 2008.

P. Fearnhead and C. Sherlock. An exact Gibbs sampler for the Markov-modulated Poisson process. *Journal Of The Royal Statistical Society Series B*, 68(5):767–784, 2006.

S. Früwirth-Schnatter. Data augmentation and dynamic linear models. *J. Time Ser. Anal.*, 15:183–202, 1994.

D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25): 2340–2361, 1977. ISSN 0022-3654.

A. Golightly and D. J. Wilkinson. Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus*, 1(6):807–820, December 2011.

A. Hobolth and E. A Stone. Simulation from endpoint-conditioned, continuous-time Markov chains on a finite state space, with applications to molecular evolution. *Ann. Appl. Stat.*, 3(3):1204, 2009. ISSN 1941-7330.

A. Jensen. Markoff chains as an aid in the study of Markoff processes. *Skand. Aktuarietiedskr.*, 36: 87–91, 1953.

O. Kallenberg. *Foundations of Modern Probability*. Probability and its Applications. Springer-Verlag, New York, Second edition, 2002. ISBN 0-387-95313-2.

C. J. Mode and G. T. Pickens. Computational methods for renewal theory and semi-Markov processes with illustrative examples. *The American Statistician*, 42(2):pp. 143–152, 1988. ISSN 00031305.

K. P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.

R. M. Neal. Slice sampling. *Annals of Statistics*, 31:705–767, 2003.

R. Nielsen. Mapping mutations on phylogenies. *Syst Biol*, 51(5):729–739, 2002.

U. Nodelman, C.R. Shelton, and D. Koller. Continuous time Bayesian networks. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 378–387, 2002.

U. Nodelman, D. Koller, and C.R. Shelton. Expectation propagation for continuous time Bayesian networks. In *Proceedings of the Twenty-first Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 431–440, July 2005.

M. Opper and G. Sanguinetti. Variational inference for Markov jump processes. In *Advances in Neural Information Processing Systems 20*, 2007.

M. Plummer, N. Best, K. Cowles, and K. Vines. CODA: Convergence diagnosis and output analysis for MCMC. *R News*, 6(1):7–11, March 2006.

V. Rao and Y. W. Teh. Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *Proceedings of the Twenty-seventh Conference on Uncertainty in Artificial Intelligence (UAI)*, 2011a.

V. Rao and Y. W. Teh. Gaussian process modulated renewal processes. In *Advances in Neural Information Processing Systems 23*, 2011b.

V. Rao and Y. W. Teh. MCMC for continuous-time discrete-state systems. In *Advances in Neural Information Processing Systems, 24*, 2012.

S. L. Scott and P. Smyth. The Markov modulated Poisson process and Markov Poisson cascade with applications to web traffic modeling. *Bayesian Statistics*, 7:1–10, 2003.

C. Shelton, Y. Fan, W. Lam, J. Lee, and J. Xu. Continuous time Bayesian network reasoning and learning engine, 2010. `http://mloss.org/software/view/216/`.

H. C. Tijms. *Stochastic Modelling and Analysis: a Computational Approach*. Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley, 1986. ISBN 9780471909118.

D. J. Wilkinson. Stochastic modelling for quantitative description of heterogeneous biological systems. *Nature Reviews Genetics*, 10:122–133, 2009.

Journal of Machine Learning Research 14 (2013) 3321-3363 Submitted 3/13; Published 11/13

Communication-Efficient Algorithms for Statistical Optimization

Yuchen Zhang YUCZHANG@EECS.BERKELEY.EDU
John C. Duchi JDUCHI@EECS.BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720 USA

Martin J. Wainwright WAINWRIG@STAT.BERKELEY.EDU
Department of Statistics
University of California, Berkeley
Berkeley, CA 94720 USA

Editor: Francis Bach

Abstract

We analyze two communication-efficient algorithms for distributed optimization in statistical settings involving large-scale data sets. The first algorithm is a standard averaging method that distributes the N data samples evenly to m machines, performs separate minimization on each subset, and then averages the estimates. We provide a sharp analysis of this average mixture algorithm, showing that under a reasonable set of conditions, the combined parameter achieves mean-squared error (MSE) that decays as $O(N^{-1}+(N/m)^{-2})$. Whenever $m \leq \sqrt{N}$, this guarantee matches the best possible rate achievable by a centralized algorithm having access to all N samples. The second algorithm is a novel method, based on an appropriate form of bootstrap subsampling. Requiring only a single round of communication, it has mean-squared error that decays as $O(N^{-1}+(N/m)^{-3})$, and so is more robust to the amount of parallelization. In addition, we show that a stochastic gradient-based method attains mean-squared error decaying as $O(N^{-1}+(N/m)^{-3/2})$, easing computation at the expense of a potentially slower MSE rate. We also provide an experimental evaluation of our methods, investigating their performance both on simulated data and on a large-scale regression problem from the internet search domain. In particular, we show that our methods can be used to efficiently solve an advertisement prediction problem from the Chinese SoSo Search Engine, which involves logistic regression with $N \approx 2.4 \times 10^8$ samples and $d \approx 740{,}000$ covariates.

Keywords: distributed learning, stochastic optimization, averaging, subsampling

1. Introduction

Many procedures for statistical estimation are based on a form of (regularized) empirical risk minimization, meaning that a parameter of interest is estimated by minimizing an objective function defined by the average of a loss function over the data. Given the current explosion in the size and amount of data available in statistical studies, a central challenge is to design efficient algorithms for solving large-scale problem instances. In a centralized setting, there are many procedures for solving empirical risk minimization problems, among them standard convex programming approaches (e.g., Boyd and Vandenberghe, 2004) as well as stochastic approximation and optimization algorithms (Robbins and Monro, 1951; Hazan et al., 2006; Nemirovski et al., 2009). When the size of the data set becomes extremely large, however, it may be infeasible to store all of the data on a

single computer, or at least to keep the data in memory. Accordingly, the focus of this paper is the study of some distributed and communication-efficient procedures for empirical risk minimization.

Recent years have witnessed a flurry of research on distributed approaches to solving very large-scale statistical optimization problems. Although we cannot survey the literature adequately—the papers Nedić and Ozdaglar (2009), Ram et al. (2010), Johansson et al. (2009), Duchi et al. (2012a), Dekel et al. (2012), Agarwal and Duchi (2011), Recht et al. (2011), Duchi et al. (2012b) and references therein contain a sample of relevant work—we touch on a few important themes here. It can be difficult within a purely optimization-theoretic setting to show explicit benefits arising from distributed computation. In statistical settings, however, distributed computation can lead to gains in computational efficiency, as shown by a number of authors (Agarwal and Duchi, 2011; Dekel et al., 2012; Recht et al., 2011; Duchi et al., 2012b). Within the family of distributed algorithms, there can be significant differences in communication complexity: different computers must be synchronized, and when the dimensionality of the data is high, communication can be prohibitively expensive. It is thus interesting to study distributed estimation algorithms that require fairly limited synchronization and communication while still enjoying the greater statistical accuracy that is usually associated with a larger data set.

With this context, perhaps the simplest algorithm for distributed statistical estimation is what we term the *average mixture* (AVGM) algorithm. It is an appealingly simple method: given m different machines and a data set of size N, first assign to each machine a (distinct) data set of size $n = N/m$, then have each machine i compute the empirical minimizer θ_i on its fraction of the data, and finally average all the parameter estimates θ_i across the machines. This approach has been studied for some classification and estimation problems by Mann et al. (2009) and McDonald et al. (2010), as well as for certain stochastic approximation methods by Zinkevich et al. (2010). Given an empirical risk minimization algorithm that works on one machine, the procedure is straightforward to implement and is extremely communication efficient, requiring only a single round of communication. It is also relatively robust to possible failures in a subset of machines and/or differences in speeds, since there is no repeated synchronization. When the local estimators are all unbiased, it is clear that the the AVGM procedure will yield an estimate that is essentially as good as that of an estimator based on all N samples. However, many estimators used in practice are biased, and so it is natural to ask whether the method has any guarantees in a more general setting. To the best of our knowledge, however, no work has shown rigorously that the AVGM procedure generally has greater efficiency than the naive approach of using $n = N/m$ samples on a single machine.

This paper makes three main contributions. First, in Section 3, we provide a sharp analysis of the AVGM algorithm, showing that under a reasonable set of conditions on the population risk, it can indeed achieve substantially better rates than the naive approach. More concretely, we provide bounds on the mean-squared error (MSE) that decay as $O((nm)^{-1} + n^{-2})$. Whenever the number of machines m is less than the number of samples n per machine, this guarantee matches the best possible rate achievable by a centralized algorithm having access to all $N = nm$ samples. In the special case of optimizing log likelihoods, the pre-factor in our bound involves the trace of the Fisher information, a quantity well-known to control the fundamental limits of statistical estimation. We also show how the result extends to stochastic programming approaches, exhibiting a stochastic gradient-descent based procedure that also attains convergence rates scaling as $O((nm)^{-1})$, but with slightly worse dependence on different problem-specific parameters.

Our second contribution is to develop a novel extension of simple averaging. It is based on an appropriate form of resampling (Efron and Tibshirani, 1993; Hall, 1992; Politis et al., 1999), which

we refer to as the *subsampled average mixture* (SAVGM) approach. At a high level, the SAVGM algorithm distributes samples evenly among m processors or computers as before, but instead of simply returning the empirical minimizer, each processor further subsamples its own data set in order to estimate the bias of its own estimate, and returns a subsample-corrected estimate. We establish that the SAVGM algorithm has mean-squared error decaying as $O(m^{-1}n^{-1}+n^{-3})$. As long as $m < n^2$, the subsampled method again matches the centralized gold standard in the first-order term, and has a second-order term smaller than the standard averaging approach.

Our third contribution is to perform a detailed empirical evaluation of both the AVGM and SAVGM procedures, which we present in Sections 4 and 5. Using simulated data from normal and non-normal regression models, we explore the conditions under which the SAVGM algorithm yields better performance than the AVGM algorithm; in addition, we study the performance of both methods relative to an oracle baseline that uses all N samples. We also study the sensitivity of the algorithms to the number of splits m of the data, and in the SAVGM case, we investigate the sensitivity of the method to the amount of resampling. These simulations show that both AVGM and SAVGM have favourable performance, even when compared to the unattainable "gold standard" procedure that has access to all N samples. In Section 5, we complement our simulation experiments with a large logistic regression experiment that arises from the problem of predicting whether a user of a search engine will click on an advertisement. This experiment is large enough—involving $N \approx 2.4 \times 10^8$ samples in $d \approx 740,000$ dimensions with a storage size of approximately 55 gigabytes—that it is difficult to solve efficiently on one machine. Consequently, a distributed approach is essential to take full advantage of this data set. Our experiments on this problem show that SAVGM—with the resampling and correction it provides—gives substantial performance benefits over naive solutions as well as the averaging algorithm AVGM.

2. Background and Problem Set-up

We begin by setting up our decision-theoretic framework for empirical risk minimization, after which we describe our algorithms and the assumptions we require for our main theoretical results.

2.1 Empirical Risk Minimization

Let $\{f(\cdot;x),\, x \in \mathcal{X}\}$ be a collection of real-valued and convex loss functions, each defined on a set containing the convex set $\Theta \subseteq \mathbb{R}^d$. Let P be a probability distribution over the sample space $\mathcal{X}$. Assuming that each function $x \mapsto f(\theta;x)$ is P-integrable, the *population risk* $F_0 : \Theta \to \mathbb{R}$ is given by

$$F_0(\theta) := \mathbb{E}_P[f(\theta;X)] = \int_{\mathcal{X}} f(\theta;x)dP(x).$$

Our goal is to estimate the parameter vector minimizing the population risk, namely the quantity

$$\theta^* := \operatorname*{argmin}_{\theta\in\Theta} F_0(\theta) = \operatorname*{argmin}_{\theta\in\Theta} \int_{\mathcal{X}} f(\theta;x)dP(x),$$

which we assume to be unique. In practice, the population distribution P is unknown to us, but we have access to a collection S of samples from the distribution P. Empirical risk minimization is based on estimating θ^* by solving the optimization problem

$$\widehat{\theta} \in \operatorname*{argmin}_{\theta\in\Theta} \Big\{ \frac{1}{|S|} \sum_{x\in S} f(\theta;x) \Big\}.$$

Throughout the paper, we impose some regularity conditions on the parameter space, the risk function F_0, and the instantaneous loss functions $f(\cdot;x) : \Theta \to \mathbb{R}$. These conditions are standard in classical statistical analysis of M-estimators (e.g., Lehmann and Casella, 1998; Keener, 2010). Our first assumption deals with the relationship of the parameter space to the optimal parameter θ^*.

Assumption 1 (Parameters) *The parameter space $\Theta \subset \mathbb{R}^d$ is a compact convex set, with $\theta^* \in \operatorname{int}\Theta$ and ℓ_2-radius $R = \max_{\theta\in\Theta} \|\theta - \theta^*\|_2$.*

In addition, the risk function is required to have some amount of curvature. We formalize this notion in terms of the Hessian of F_0:

Assumption 2 (Local strong convexity) *The population risk is twice differentiable, and there exists a parameter $\lambda > 0$ such that $\nabla^2 F_0(\theta^*) \succeq \lambda I_{d\times d}$.*

Here $\nabla^2 F_0(\theta)$ denotes the $d \times d$ Hessian matrix of the population objective F_0 evaluated at θ, and we use $\succeq$ to denote the positive semidefinite ordering (i.e., $A \succeq B$ means that $A - B$ is positive semidefinite.) This local condition is milder than a global strong convexity condition and is required to hold only for the population risk F_0 evaluated at θ^*. It is worth observing that some type of curvature of the risk is required for any method to consistently estimate the parameters θ^*.

2.2 Averaging Methods

Consider a data set consisting of $N = mn$ samples, drawn i.i.d. according to the distribution P. In the distributed setting, we divide this N-sample data set evenly and uniformly at random among a total of m processors. (For simplicity, we have assumed the total number of samples is a multiple of m.) For $i = 1, \ldots, m$, we let $S_{1,i}$ denote the data set assigned to processor i; by construction, it is a collection of n samples drawn i.i.d. according to P, and the samples in subsets $S_{1,i}$ and $S_{1,j}$ are independent for $i \neq j$. In addition, for each processor i we define the (local) empirical distribution $P_{1,i}$ and empirical objective $F_{1,i}$ via

$$P_{1,i} := \frac{1}{|S_1|} \sum_{x\in S_{1,i}} \delta_x, \quad \text{and} \quad F_{1,i}(\theta) := \frac{1}{|S_{1,i}|} \sum_{x\in S_{1,i}} f(\theta; x).$$

With this notation, the AVGM algorithm is very simple to describe.

2.2.1 AVERAGE MIXTURE ALGORITHM

(1) For each $i \in \{1, \ldots, m\}$, processor i uses its local data set $S_{1,i}$ to compute the local empirical minimizer

$$\theta_{1,i} \in \operatorname*{argmin}_{\theta\in\Theta} \Big\{ \underbrace{\frac{1}{|S_{1,i}|} \sum_{x\in S_{1,i}} f(\theta; x)}_{F_{1,i}(\theta)} \Big\}. \tag{1}$$

(2) These m local estimates are then averaged together—that is, we compute

$$\bar{\theta}_1 = \frac{1}{m} \sum_{i=1}^{m} \theta_{1,i}. \tag{2}$$

The subsampled average mixture (SAVGM) algorithm is based on an additional level of sampling on top of the first, involving a fixed subsampling rate $r \in [0,1]$. It consists of the following additional steps:

2.2.2 SUBSAMPLED AVERAGE MIXTURE ALGORITHM

(1) Each processor i draws a subset $S_{2,i}$ of size $\lceil rn \rceil$ by sampling uniformly at random without replacement from its local data set $S_{1,i}$.

(2) Each processor i computes both the local empirical minimizers $\theta_{1,i}$ from Equation (1) and the empirical minimizer

$$\theta_{2,i} \in \operatorname*{argmin}_{\theta \in \Theta} \Big\{ \underbrace{\frac{1}{|S_{2,i}|} \sum_{x \in S_{2,i}} f(\theta; x)}_{F_{2,i}(\theta)} \Big\}.$$

(3) In addition to the previous average (2), the SAVGM algorithm computes the bootstrap average $\overline{\theta}_2 := \frac{1}{m} \sum_{i=1}^m \theta_{2,i}$, and then returns the weighted combination

$$\overline{\theta}_{\text{SAVGM}} := \frac{\overline{\theta}_1 - r\overline{\theta}_2}{1-r}. \tag{3}$$

The intuition for the weighted estimator (3) is similar to that for standard bias correction procedures using the bootstrap or subsampling (Efron and Tibshirani, 1993; Hall, 1992; Politis et al., 1999). Roughly speaking, if $b_0 = \theta^* - \theta_1$ is the bias of the first estimator, then we may approximate b_0 by the subsampled estimate of bias $b_1 = \theta^* - \theta_2$. Then, we use the fact that $b_1 \approx b_0/r$ to argue that $\theta^* \approx (\theta_1 - r\theta_2)/(1-r)$. The re-normalization enforces that the relative "weights" of $\overline{\theta}_1$ and $\overline{\theta}_2$ sum to 1.

The goal of this paper is to understand under what conditions—and in what sense—the estimators (2) and (3) approach the *oracle performance*, by which we mean the error of a centralized risk minimization procedure that is given access to all $N = nm$ samples.

2.2.3 NOTATION

Before continuing, we define the remainder of our notation. We use ℓ_2 to denote the usual Euclidean norm $\|\theta\|_2 = (\sum_{j=1}^d \theta_j^2)^{\frac{1}{2}}$. The ℓ_2-operator norm of a matrix $A \in \mathbb{R}^{d_1 \times d_2}$ is its maximum singular value, defined by

$$|||A|||_2 := \sup_{v \in \mathbb{R}^{d_2}, \|v\|_2 \leq 1} \|Av\|_2.$$

A convex function F is λ-strongly convex on a set $U \subseteq \mathbb{R}^d$ if for arbitrary $u, v \in U$ we have

$$F(u) \geq F(v) + \langle \nabla F(v), u - v \rangle + \frac{\lambda}{2} \|u - v\|_2^2.$$

(If F is not differentiable, we may replace ∇F with any subgradient of F.) We let $\otimes$ denote the Kronecker product, and for a pair of vectors u, v, we define the outer product $u \otimes v = uv^\top$. For a

three-times differentiable function F, we denote the third derivative tensor by $\nabla^3 F$, so that for each $u \in \operatorname{dom} F$ the operator $\nabla^3 F(u) : \mathbb{R}^{d \times d} \to \mathbb{R}^d$ is linear and satisfies the relation

$$\left[\nabla^3 F(u)(v \otimes v)\right]_i = \sum_{j,k=1}^{d} \left(\frac{\partial^3}{\partial u_i \partial u_j \partial u_k} F(u) \right) v_j v_k.$$

We denote the indicator function of an event $\mathcal{E}$ by $1_{(\mathcal{E})}$, which is 1 if $\mathcal{E}$ is true and 0 otherwise.

3. Theoretical Results

Having described the AVGM and SAVGM algorithms, we now turn to statements of our main theorems on their statistical properties, along with some consequences and comparison to past work.

3.1 Smoothness Conditions

In addition to our previously stated assumptions on the population risk, we require regularity conditions on the empirical risk functions. It is simplest to state these in terms of the functions $\theta \mapsto f(\theta; x)$, and we note that, as with Assumption 2, we require these to hold only locally around the optimal point θ^*, in particular within some Euclidean ball $U = \{\theta \in \mathbb{R}^d \mid \|\theta^* - \theta\|_2 \le \rho\} \subseteq \Theta$ of radius $\rho > 0$.

Assumption 3 (Smoothness) *There are finite constants G, H such that the first and the second partial derivatives of f exist and satisfy the bounds*

$$\mathbb{E}[\|\nabla f(\theta; X)\|_2^8] \le G^8 \quad \text{and} \quad \mathbb{E}[\left|\left|\left|\nabla^2 f(\theta; X) - \nabla^2 F_0(\theta)\right|\right|\right|_2^8] \le H^8 \quad \text{for all } \theta \in U.$$

In addition, for any $x \in \mathcal{X}$, the Hessian matrix $\nabla^2 f(\theta; x)$ is $L(x)$-Lipschitz continuous, meaning that

$$\left|\left|\left|\nabla^2 f(\theta'; x) - \nabla^2 f(\theta; x)\right|\right|\right|_2 \le L(x) \left\|\theta' - \theta\right\|_2 \quad \text{for all } \theta, \theta' \in U. \tag{4}$$

We require that $\mathbb{E}[L(X)^8] \le L^8$ and $\mathbb{E}[(L(X) - \mathbb{E}[L(X)])^8] \le L^8$ for some finite constant L.

It is an important insight of our analysis that some type of smoothness condition on the Hessian matrix, as in the Lipschitz condition (4), is *essential* in order for simple averaging methods to work. This necessity is illustrated by the following example:

Example 1 (Necessity of Hessian conditions) *Let X be a Bernoulli variable with parameter $\frac{1}{2}$, and consider the loss function*

$$f(\theta; x) = \begin{cases} \theta^2 - \theta & \text{if } x = 0 \\ \theta^2 1_{(\theta \le 0)} + \theta & \text{if } x = 1, \end{cases} \tag{5}$$

where $1_{(\theta \le 0)}$ is the indicator of the event $\{\theta \le 0\}$. The associated population risk is $F_0(\theta) = \frac{1}{2}(\theta^2 + \theta^2 1_{(\theta \le 0)})$. Since $|F_0'(w) - F_0'(v)| \le 2|w - v|$, the population risk is strongly convex and smooth, but it has discontinuous second derivative. The unique minimizer of the population risk is $\theta^* = 0$, and by an asymptotic expansion given in Appendix A, it can be shown that $\mathbb{E}[\theta_{1,i}] = \Omega(n^{-\frac{1}{2}})$. Consequently, the bias of $\overline{\theta}_1$ is $\Omega(n^{-\frac{1}{2}})$, and the AVGM algorithm using $N = mn$ observations must suffer mean squared error $\mathbb{E}[(\overline{\theta}_1 - \theta^*)^2] = \Omega(n^{-1})$.

The previous example establishes the necessity of a smoothness condition. However, in a certain sense, it is a pathological case: both the smoothness condition given in Assumption 3 and the local strong convexity condition given in Assumption 2 are relatively innocuous for practical problems. For instance, both conditions will hold for standard forms of regression, such as linear and logistic, as long as the *population* data covariance matrix is not rank deficient and the data has suitable moments. Moreover, in the linear regression case, one has $L = 0$.

3.2 Bounds for Simple Averaging

We now turn to our first theorem that provides guarantees on the statistical error associated with the AVGM procedure. We recall that θ^* denotes the minimizer of the population objective function F_0, and that for each $i \in \{1, \ldots, m\}$, we use S_i to denote a data set of n independent samples. For each i, we use $\theta_i \in \operatorname{argmin}_{\theta \in \Theta}\{\frac{1}{n}\sum_{x \in S_i} f(\theta; x)\}$ to denote a minimizer of the empirical risk for the data set S_i, and we define the averaged vector $\overline{\theta} = \frac{1}{m}\sum_{i=1}^m \theta_i$. The following result bounds the mean-squared error between this averaged estimate and the minimizer θ^* of the population risk.

Theorem 1 *Under Assumptions 1 through 3, the mean-squared error is upper bounded as*

$$\begin{aligned} \mathbb{E}\left[\left\|\overline{\theta} - \theta^*\right\|_2^2\right] &\leq \frac{2}{nm}\mathbb{E}\left[\left\|\nabla^2 F_0(\theta^*)^{-1}\nabla f(\theta^*; X)\right\|_2^2\right] \\ &\quad + \frac{c}{\lambda^2 n^2}\left(H^2 \log d + \frac{L^2 G^2}{\lambda^2}\right)\mathbb{E}\left[\left\|\nabla^2 F_0(\theta^*)^{-1}\nabla f(\theta^*; X)\right\|_2^2\right] \\ &\quad + O(m^{-1}n^{-2}) + O(n^{-3}), \end{aligned} \tag{6}$$

where c is a numerical constant.

A slightly weaker corollary of Theorem 1 makes it easier to parse. In particular, note that

$$\left\|\nabla^2 F_0(\theta^*)^{-1}\nabla f(\theta^*; x)\right\|_2 \overset{(i)}{\leq} \left|\left|\left|\nabla^2 F_0(\theta^*)^{-1}\right|\right|\right|_2 \|\nabla f(\theta^*; x)\|_2 \overset{(ii)}{\leq} \frac{1}{\lambda}\|\nabla f(\theta^*; x)\|_2, \tag{7}$$

where step (i) follows from the inequality $|||Ax|||_2 \leq |||A|||\,\|x\|_2$, valid for any matrix A and vector x; and step (ii) follows from Assumption 2. In addition, Assumption 3 implies $\mathbb{E}[\|\nabla f(\theta^*; X)\|_2^2] \leq G^2$, and putting together the pieces, we have established the following.

Corollary 2 *Under the same conditions as Theorem 1,*

$$\mathbb{E}\left[\left\|\overline{\theta} - \theta^*\right\|_2^2\right] \leq \frac{2G^2}{\lambda^2 nm} + \frac{cG^2}{\lambda^4 n^2}\left(H^2 \log d + \frac{L^2 G^2}{\lambda^2}\right) + O(m^{-1}n^{-2}) + O(n^{-3}). \tag{8}$$

This upper bound shows that the leading term decays proportionally to $(nm)^{-1}$, with the pre-factor depending inversely on the strong convexity constant λ and growing proportionally with the bound G on the loss gradient. Although easily interpretable, the upper bound (8) can be loose, since it is based on the relatively weak series of bounds (7).

The leading term in our original upper bound (6) involves the product of the gradient $\nabla f(\theta^*; X)$ with the inverse Hessian. In many statistical settings, including the problem of linear regression, the effect of this matrix-vector multiplication is to perform some type of standardization. When the

loss $f(\cdot;x) : \Theta \to \mathbb{R}$ is actually the negative log-likelihood $\ell(x \mid \theta)$ for a parametric family of models $\{P_\theta\}$, we can make this intuition precise. In particular, under suitable regularity conditions (e.g., Lehmann and Casella, 1998, Chapter 6), we can define the Fisher information matrix

$$I(\theta^*) := \mathbb{E}\left[\nabla\ell(X \mid \theta^*)\nabla\ell(X \mid \theta^*)^\top\right] = \mathbb{E}[\nabla^2\ell(X \mid \theta^*)].$$

Recalling that $N = mn$ is the total number of samples available, let us define the neighbourhood $B_2(\theta, t) := \{\theta' \in \mathbb{R}^d : \|\theta' - \theta\|_2 \le t\}$. Then under our assumptions, the Hájek-Le Cam minimax theorem (van der Vaart, 1998, Theorem 8.11) guarantees for *any estimator* $\widehat{\theta}_N$ based on N samples that

$$\lim_{c\to\infty} \liminf_{N\to\infty} \sup_{\theta \in B_2(\theta^*, c/\sqrt{N})} N\,\mathbb{E}_\theta\left[\left\|\widehat{\theta}_N - \theta\right\|_2^2\right] \ge \operatorname{tr}(I(\theta^*)^{-1}).$$

In connection with Theorem 1, we obtain:

Corollary 3 *In addition to the conditions of Theorem 1, suppose that the loss functions $f(\cdot;x)$ are the negative log-likelihood $\ell(x \mid \theta)$ for a parametric family $\{P_\theta, \theta \in \Theta\}$. Then the mean-squared error is upper bounded as*

$$\mathbb{E}\left[\left\|\overline{\theta}_1 - \theta^*\right\|_2^2\right] \le \frac{2}{N}\operatorname{tr}(I(\theta^*)^{-1}) + \frac{cm^2\operatorname{tr}(I(\theta^*)^{-1})}{\lambda^2 N^2}\left(H^2\log d + \frac{L^2G^2}{\lambda^2}\right) + O(mN^{-2}),$$

where c is a numerical constant.

Proof Rewriting the log-likelihood in the notation of Theorem 1, we have $\nabla\ell(x \mid \theta^*) = \nabla f(\theta^*;x)$ and all we need to note is that

$$\begin{aligned} I(\theta^*)^{-1} &= \mathbb{E}\left[I(\theta^*)^{-1}\nabla\ell(X \mid \theta^*)\nabla\ell(X \mid \theta^*)^\top I(\theta^*)^{-1}\right] \\ &= \mathbb{E}\left[\left(\nabla^2 F_0(\theta^*)^{-1}\nabla f(\theta^*;X)\right)\left(\nabla^2 F_0(\theta^*)^{-1}\nabla f(\theta^*;X)\right)^\top\right]. \end{aligned}$$

Now apply the linearity of the trace and use the fact that $\operatorname{tr}(uu^\top) = \|u\|_2^2$. ■

Except for the factor of two in the bound, Corollary 3 shows that Theorem 1 essentially achieves the best possible result. The important aspect of our bound, however, is that we obtain this convergence rate without calculating an estimate on all $N = mn$ samples: instead, we calculate m independent estimators, and then average them to attain the convergence guarantee. We remark that an inspection of our proof shows that, at the expense of worse constants on higher order terms, we can reduce the factor of $2/mn$ on the leading term in Theorem 1 to $(1+c)/mn$ for any constant $c > 0$; as made clear by Corollary 3, this is unimprovable, even by constant factors.

As noted in the introduction, our bounds are certainly to be expected for unbiased estimators, since in such cases averaging m independent solutions reduces the variance by $1/m$. In this sense, our results are similar to classical distributional convergence results in M-estimation: for smooth enough problems, M-estimators behave asymptotically like averages (van der Vaart, 1998; Lehmann and Casella, 1998), and averaging multiple independent realizations reduces their variance. However, it is often desirable to use biased estimators, and such bias introduces difficulty in the analysis, which we explore more in the next section. We also note that in contrast to classical asymptotic results, our results are applicable to finite samples and give explicit upper bounds on the mean-squared error. Lastly, our results are not tied to a specific model, which allows for fairly general sampling distributions.

3.3 Bounds for Subsampled Mixture Averaging

When the number of machines m is relatively small, Theorem 1 and Corollary 2 show that the convergence rate of the AVGM algorithm is mainly determined by the first term in the bound (6), which is at most $\frac{G^2}{\lambda^2 mn}$. In contrast, when the number of processors m grows, the second term in the bound (6), in spite of being $O(n^{-2})$, may have non-negligible effect. This issue is exacerbated when the local strong convexity parameter λ of the risk F_0 is close to zero or the Lipschitz continuity constant H of ∇f is large. This concern motivated our development of the subsampled average mixture (SAVGM) algorithm, to which we now return.

Due to the additional randomness introduced by the subsampling in SAVGM, its analysis requires an additional smoothness condition. In particular, recalling the Euclidean ρ-neighbourhood U of the optimum θ^*, we require that the loss function f is (locally) smooth through its third derivatives.

Assumption 4 (Strong smoothness) *For each $x \in \mathcal{X}$, the third derivatives of f are $M(x)$-Lipschitz continuous, meaning that*

$$\left\| \left(\nabla^3 f(\theta; x) - \nabla^3 f(\theta'; x) \right) (u \otimes u) \right\|_2 \leq M(x) \left\| \theta - \theta' \right\|_2 \|u\|_2^2 \quad \text{for all } \theta, \theta' \in U, \text{ and } u \in \mathbb{R}^d,$$

where $\mathbb{E}[M^8(X)] \leq M^8$ for some constant $M < \infty$.

It is easy to verify that Assumption 4 holds for least-squares regression with $M = 0$. It also holds for various types of non-linear regression problems (e.g., logistic, multinomial etc.) as long as the covariates have finite eighth moments.

With this set-up, our second theorem establishes that bootstrap sampling yields improved performance:

Theorem 4 *Under Assumptions 1 through 4, the output $\overline{\theta}_{\text{SAVGM}} = (\overline{\theta}_1 - r\overline{\theta}_2)/(1-r)$ of the bootstrap* SAVGM *algorithm has mean-squared error bounded as*

$$\mathbb{E}\left[\left\| \overline{\theta}_{\text{SAVGM}} - \theta^* \right\|_2^2 \right] \leq \frac{2+3r}{(1-r)^2} \cdot \frac{1}{nm} \mathbb{E}\left[\left\| \nabla^2 F_0(\theta^*)^{-1} \nabla f(\theta^*; X) \right\|_2^2 \right] + c\left(\frac{M^2 G^6}{\lambda^6} + \frac{G^4 L^2 d \log d}{\lambda^4} \right) \left(\frac{1}{r(1-r)^2} \right) n^{-3} + O\left(\frac{1}{(1-r)^2} m^{-1} n^{-2} \right) \tag{9}$$

for a numerical constant c.

Comparing the conclusions of Theorem 4 to those of Theorem 1, we see that the the $O(n^{-2})$ term in the bound (6) has been eliminated. The reason for this elimination is that subsampling at a rate r reduces the bias of the SAVGM algorithm to $O(n^{-3})$, whereas in contrast, the bias of the AVGM algorithm induces terms of order n^{-2}. Theorem 4 suggests that the performance of the SAVGM algorithm is affected by the subsampling rate r; in order to minimize the upper bound (9) in the regime $m < N^{2/3}$, the optimal choice is of the form $r \propto C\sqrt{m}/n = Cm^{3/2}/N$ where $C \approx (G^2/\lambda^2) \max\{MG/\lambda, L\sqrt{d \log d}\}$. Roughly, as the number of machines m becomes larger, we may increase r, since we enjoy averaging affects from the SAVGM algorithm.

Let us consider the relative effects of having larger numbers of machines m for both the AVGM and SAVGM algorithms, which provides some guidance to selecting m in practice. We define $\sigma^2 =$

$\mathbb{E}[\left\|\nabla^2 F_0(\theta^*)^{-1}\nabla f(\theta^*;X)\right\|_2^2]$ to be the asymptotic variance. Then to obtain the optimal convergence rate of σ^2/N, we must have

$$\frac{1}{\lambda^2}\max\left\{H^2\log d, L^2G^2\right\}\frac{m^2}{N^2}\sigma^2 \leq \frac{\sigma^2}{N} \quad \text{or} \quad m \leq N^{\frac{1}{2}}\sqrt{\frac{\lambda^2}{\max\{H^2\log d, L^2G^2/\lambda^2\}}} \tag{10}$$

in Theorem 1. Applying the bound of Theorem 4, we find that to obtain the same rate we require

$$\max\left\{\frac{M^2G^2}{\lambda^6}, \frac{L^2 d\log d}{\lambda^4}\right\}\frac{G^4m^3}{rN^3} \leq \frac{(1+r)\sigma^2}{N} \text{ or } m \leq N^{\frac{2}{3}}\left(\frac{\lambda^4 r(1+r)\sigma^2}{\max\{M^2G^6/\lambda^2, G^4L^2 d\log d\}}\right)^{\frac{1}{3}}.$$

Now suppose that we replace r with $Cm^{3/2}/N$ as in the previous paragraph. Under the conditions $\sigma^2 \approx G^2$ and $r = o(1)$, we then find that

$$m \leq N^{\frac{2}{3}}\left(\frac{\lambda^2\sigma^2 m^{3/2}}{G^2\max\left\{MG/\lambda, L\sqrt{d\log d}\right\}N}\right)^{\frac{1}{3}} \text{ or } m \leq N^{\frac{2}{3}}\left(\frac{\lambda^2}{\max\left\{MG/\lambda, L\sqrt{d\log d}\right\}}\right)^{\frac{2}{3}}. \tag{11}$$

Comparing inequalities (10) and (11), we see that in both cases m may grow polynomially with the global sample size N while still guaranteeing optimal convergence rates. On one hand, this asymptotic growth is faster in the subsampled case (11); on the other hand, the dependence on the dimension d of the problem is more stringent than the standard averaging case (10). As the local strong convexity constant λ of the *population risk* shrinks, both methods allow less splitting of the data, meaning that the sample size per machine must be larger. This limitation is intuitive, since lower curvature for the population risk means that the local empirical risks associated with each machine will inherit lower curvature as well, and this effect will be exacerbated with a small local sample size per machine. Averaging methods are, of course, not a panacea: the allowed number of partitions m does not grow linearly in either case, so blindly increasing the number of machines proportionally to the total sample size N will not lead to a useful estimate.

In practice, an optimal choice of r may not be apparent, which may necessitate cross validation or another type of model evaluation. We leave as intriguing open questions whether computing multiple subsamples at each machine can yield improved performance or reduce the variance of the SAVGM procedure, and whether using estimates based on resampling the data with replacement, as opposed to without replacement as considered here, can yield improved performance.

3.4 Time Complexity

In practice, the exact empirical minimizers assumed in Theorems 1 and 4 may be unavailable. Instead, we need to use a finite number of iterations of some optimization algorithm in order to obtain reasonable approximations to the exact minimizers. In this section, we sketch an argument that shows that both the AVGM algorithm and the SAVGM algorithm can use such approximate empirical minimizers, and as long as the optimization error is sufficiently small, the resulting averaged estimate achieves the same order-optimal statistical error. Here we provide the arguments only for the AVGM algorithm; the arguments for the SAVGM algorithm are analogous.

More precisely, suppose that each processor runs a finite number of iterations of some optimization algorithm, thereby obtaining the vector θ_i' as an approximate minimizer of the objective function $F_{1,i}$. Thus, the vector θ_i' can be viewed as an approximate form of θ_i, and we let $\overline{\theta}' = \frac{1}{m}\sum_{i=1}^m \theta_i'$ denote

the average of these approximate minimizers, which corresponds to the output of the approximate AVGM algorithm. With this notation, we have

$$\mathbb{E}\left[\left\|\overline{\theta}' - \theta^*\right\|_2^2\right] \overset{(i)}{\leq} 2\mathbb{E}[\left\|\overline{\theta} - \theta^*\right\|_2^2] + 2\mathbb{E}\left[\left\|\overline{\theta}' - \overline{\theta}\right\|_2^2\right] \overset{(ii)}{\leq} 2\mathbb{E}[\left\|\overline{\theta} - \theta^*\right\|_2^2] + 2\mathbb{E}[\left\|\theta_1' - \theta_1\right\|_2^2], \quad (12)$$

where step (i) follows by triangle inequality and the elementary bound $(a+b)^2 \leq 2a^2 + 2b^2$; step (ii) follows by Jensen's inequality. Consequently, suppose that processor i runs enough iterations to obtain an approximate minimizer θ_1' such that

$$\mathbb{E}[\left\|\theta_i' - \theta_i\right\|_2^2] = O((mn)^{-2}). \quad (13)$$

When this condition holds, the bound (12) shows that the average $\overline{\theta}'$ of the approximate minimizers shares the same convergence rates provided by Theorem 1.

But how long does it take to compute an approximate minimizer θ_i' satisfying condition (13)? Assuming processing one sample requires one unit of time, we claim that this computation can be performed in time $O(n\log(mn))$. In particular, the following two-stage strategy, involving a combination of stochastic gradient descent (see the following subsection for more details) and standard gradient descent, has this complexity:

(1) As shown in the proof of Theorem 1, with high probability, the empirical risk F_1 is strongly convex in a ball $B_\rho(\theta_1)$ of constant radius $\rho > 0$ around θ_1. Consequently, performing stochastic gradient descent on F_1 for $O(\log^2(mn)/\rho^2)$ iterations yields an approximate minimizer that falls within $B_\rho(\theta_1)$ with high probability (e.g., Nemirovski et al., 2009, Proposition 2.1). Note that the radius ρ for local strong convexity is a property of the population risk F_0 we use as a prior knowledge.

(2) This initial estimate can be further improved by a few iterations of standard gradient descent. Under local strong convexity of the objective function, gradient descent is known to converge at a geometric rate (see, e.g., Nocedal and Wright, 2006; Boyd and Vandenberghe, 2004), so $O(\log(1/\varepsilon))$ iterations will reduce the error to order ε. In our case, we have $\varepsilon = (mn)^{-2}$, and since each iteration of standard gradient descent requires $O(n)$ units of time, a total of $O(n\log(mn))$ time units are sufficient to obtain a final estimate θ_1' satisfying condition (13).

Overall, we conclude that the speed-up of the AVGM relative to the naive approach of processing all $N = mn$ samples on one processor, is at least of order $m/\log(N)$.

3.5 Stochastic Gradient Descent with Averaging

The previous strategy involved a combination of stochastic gradient descent and standard gradient descent. In many settings, it may be appealing to use only a stochastic gradient algorithm, due to their ease of their implementation and limited computational requirements. In this section, we describe an extension of Theorem 1 to the case in which each machine computes an approximate minimizer using only stochastic gradient descent.

Stochastic gradient algorithms have a lengthy history in statistics, optimization, and machine learning (Robbins and Monro, 1951; Polyak and Juditsky, 1992; Nemirovski et al., 2009; Rakhlin et al., 2012). Let us begin by briefly reviewing the basic form of stochastic gradient descent (SGD). Stochastic gradient descent algorithms iteratively update a parameter vector θ^t over time based on

randomly sampled gradient information. Specifically, at iteration t, a sample X_t is drawn at random from the distribution P (or, in the case of a finite set of data $\{X_1, \ldots, X_n\}$, a sample X_t is chosen from the data set). The method then performs the following two steps:

$$\theta^{t+\frac{1}{2}} = \theta^t - \eta_t \nabla f(\theta^t; X_t) \quad \text{and} \quad \theta^{t+1} = \operatorname*{argmin}_{\theta \in \Theta} \left\{ \left\| \theta - \theta^{t+\frac{1}{2}} \right\|_2^2 \right\}. \tag{14}$$

Here $\eta_t > 0$ is a stepsize, and the first update in (14) is a gradient descent step with respect to the random gradient $\nabla f(\theta^t; X_t)$. The method then projects the intermediate point $\theta^{t+\frac{1}{2}}$ back onto the constraint set Θ (if there is a constraint set). The convergence of SGD methods of the form (14) has been well-studied, and we refer the reader to the papers by Polyak and Juditsky (1992), Nemirovski et al. (2009), and Rakhlin et al. (2012) for deeper investigations.

To prove convergence of our stochastic gradient-based averaging algorithms, we require the following smoothness and strong convexity condition, which is an alternative to the Assumptions 2 and 3 used previously.

Assumption 5 (Smoothness and Strong Convexity II) *There exists a function $L : \mathcal{X} \to \mathbb{R}_+$ such that*

$$\left\| \left\| \nabla^2 f(\theta; x) - \nabla^2 f(\theta^*; x) \right\| \right\|_2 \leq L(x) \left\| \theta - \theta^* \right\|_2 \quad \textit{for all } x \in \mathcal{X},$$

and $\mathbb{E}[L^2(X)] \leq L^2 < \infty$. There are finite constants G and H such that

$$\mathbb{E}[\|\nabla f(\theta; X)\|_2^4] \leq G^4, \quad \textit{and} \quad \mathbb{E}[\left\| \left\| \nabla^2 f(\theta^*; X) \right\| \right\|_2^4] \leq H^4 \quad \textit{for each fixed } \theta \in \Theta.$$

In addition, the population function F_0 is λ-strongly convex over the space Θ, meaning that

$$\nabla^2 F_0(\theta) \succeq \lambda I_{d \times d} \quad \textit{for all } \theta \in \Theta.$$

Assumption 5 does not require as many moments as does Assumption 3, but it does require each moment bound to hold globally, that is, over the entire space Θ, rather than only in a neighbourhood of the optimal point θ^*. Similarly, the necessary curvature—in the form of the lower bound on the Hessian matrix $\nabla^2 F_0$—is also required to hold globally, rather than only locally. Nonetheless, Assumption 5 holds for many common problems; for instance, it holds for any linear regression problem in which the covariates have finite fourth moments and the domain Θ is compact.

The averaged stochastic gradient algorithm (SGDAVGM) is based on the following two steps:

(1) Given some constant $c > 1$, each machine performs n iterations of stochastic gradient descent (14) on its local data set of n samples using the stepsize $\eta_t = \frac{c}{\lambda t}$, then outputs the resulting local parameter θ_i'.

(2) The algorithm computes the average $\overline{\theta}^n = \frac{1}{m} \sum_{i=1}^m \theta_i'$.

The following result characterizes the mean-squared error of this procedure in terms of the constants

$$\alpha := 4c^2 \quad \text{and} \quad \beta := \max \left\{ \left\lceil \frac{cH}{\lambda} \right\rceil, \frac{c\alpha^{3/4} G^{3/2}}{(c-1)\lambda^{5/2}} \left(\frac{\alpha^{1/4} L G^{1/2}}{\lambda^{1/2}} + \frac{4G + HR}{\rho^{3/2}} \right) \right\}.$$

Theorem 5 *Under Assumptions 1 and 5, the output $\bar{\theta}^n$ of the* SAVGM *algorithm has mean-squared error upper bounded as*

$$\mathbb{E}\left[\left\|\bar{\theta}^n - \theta^*\right\|_2^2\right] \leq \frac{\alpha G^2}{\lambda^2 mn} + \frac{\beta^2}{n^{3/2}}. \tag{15}$$

Theorem 5 shows that the averaged stochastic gradient descent procedure attains the optimal convergence rate $O(N^{-1})$ as a function of the total number of observations $N = mn$. The constant and problem-dependent factors are somewhat worse than those in the earlier results we presented in Theorems 1 and 4, but the practical implementability of such a procedure may in some circumstances outweigh those differences. We also note that the second term of order $O(n^{-3/2})$ may be reduced to $O(n^{(2-2k)/k})$ for any $k \geq 4$ by assuming the existence of kth moments in Assumption 5; we show this in passing after our proof of the theorem in Appendix D. It is not clear whether a bootstrap correction is possible for the stochastic-gradient based estimator; such a correction could be significant, because the term $\beta^2/n^{3/2}$ arising from the bias in the stochastic gradient estimator may be non-trivial. We leave this question to future work.

4. Performance on Synthetic Data

In this section, we report the results of simulation studies comparing the AVGM, SAVGM, and SGDAVGM methods, as well as a trivial method using only a fraction of the data available on a single machine. For each of our simulated experiments, we use a fixed total number of samples $N = 100{,}000$, but we vary the number of parallel splits m of the data (and consequently, the local data set sizes $n = N/m$) and the dimensionality d of the problem solved.

For our experiments, we simulate data from one of three regression models:

$$y = \langle u, x\rangle + \varepsilon, \tag{16}$$

$$y = \langle u, x\rangle + \sum_{j=1}^{d} v_j x_j^3 + \varepsilon, \quad \text{or} \tag{17}$$

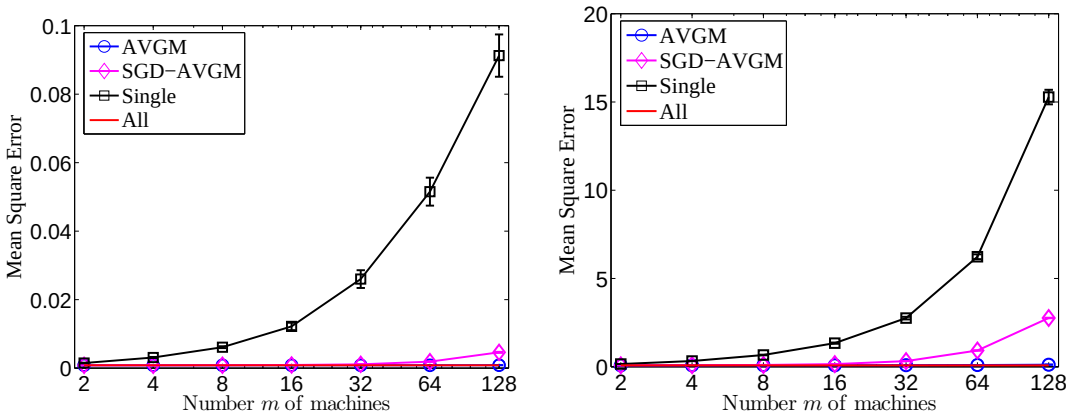
$$y = \langle u, x\rangle + h(x)|\varepsilon|, \tag{18}$$

where $\varepsilon \sim N(0,1)$, and h is a function to be specified. Specifically, the data generation procedure is as follows. For each individual simulation, we choose fixed vector $u \in \mathbb{R}^d$ with entries u_i distributed uniformly in $[0,1]$ (and similarly for v), and we set $h(x) = \sum_{j=1}^{d}(x_j/2)^3$. The models (16) through (18) provide points on a curve from correctly-specified to grossly mis-specified models, so models (17) and (18) help us understand the effects of subsampling in the SAVGM algorithm. (In contrast, the standard least-squares estimator is unbiased for model (16).) The noise variable ε is always chosen as a standard Gaussian variate $N(0,1)$, independent from sample to sample.

In our simulation experiments we use the least-squares loss

$$f(\theta; (x,y)) := \frac{1}{2}(\langle \theta, x\rangle - y)^2.$$

The goal in each experiment is to estimate the vector θ^* minimizing $F_0(\theta) := \mathbb{E}[f(\theta; (X,Y))]$. For each simulation, we generate N samples according to either the model (16) or (18). For each $m \in \{2, 4, 8, 16, 32, 64, 128\}$, we estimate $\theta^* = \arg\min_\theta F_0(\theta)$ using a parallel method with data split into m independent sets of size $n = N/m$, specifically



(a) $d = 20$ (b) $d = 200$

Figure 1: The error $\|\widehat{\theta} - \theta^*\|_2^2$ versus number of machines, with standard errors across twenty simulations, for solving least squares with data generated according to the normal model (16). The oracle least-squares estimate using all N samples is given by the line "All," while the line "Single" gives the performance of the naive estimator using only $n = N/m$ samples.

(i) The AVGM method

(ii) The SAVGM method with several settings of the subsampling ratio r

(iii) The SGDAVGM method with stepsize $\eta_t = d/(10(d+t))$, which gave good performance.

In addition to (i)–(iii), we also estimate θ^* with

(iv) The empirical minimizer of a single split of the data of size $n = N/m$

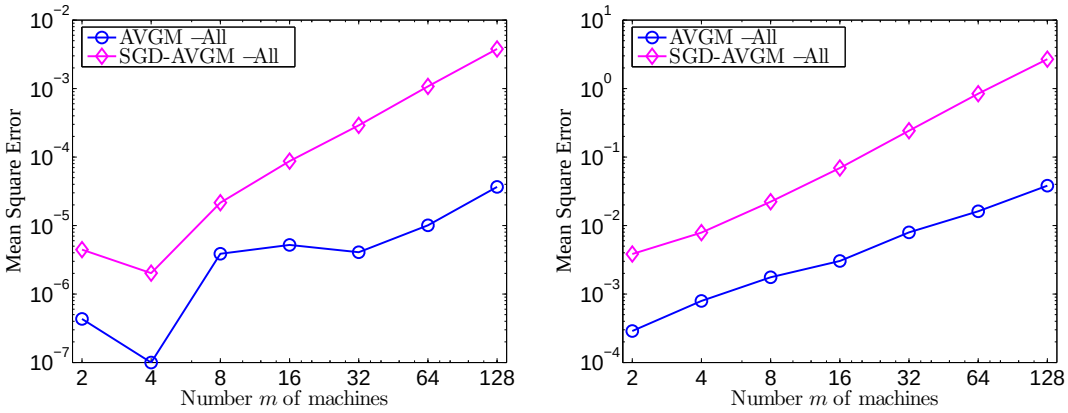
(v) The empirical minimizer on the full data set (the oracle solution).

4.1 Averaging Methods

For our first set of experiments, we study the performance of the averaging methods (AVGM and SAVGM), showing their scaling as the number of splits of data—the number of machines m—grows for fixed N and dimensions $d = 20$ and $d = 200$. We use the standard regression model (16) to generate the data, and throughout we let $\widehat{\theta}$ denote the estimate returned by the method under consideration (so in the AVGM case, for example, this is the vector $\widehat{\theta} := \overline{\theta}_1$). The data samples consist of pairs (x, y), where $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$ is the target value. To sample each x vector, we choose five distinct indices in $\{1, \ldots, d\}$ uniformly at random, and the entries of x at those indices are distributed as $N(0,1)$. For the model (16), the population optimal vector θ^* is u.

In Figure 1, we plot the error $\|\widehat{\theta} - \theta^*\|_2^2$ of the inferred parameter vector $\widehat{\theta}$ for the true parameters θ^* versus the number of splits m, or equivalently, the number of separate machines available for use. We also plot standard errors (across twenty experiments) for each curve. As a baseline in each plot, we plot as a red line the squared error $\|\widehat{\theta}_N - \theta^*\|_2^2$ of the centralized "gold standard," obtained by applying a batch method to all N samples.

From the plots in Figure 1, we can make a few observations. The AVGM algorithm enjoys excellent performance, as predicted by our theoretical results, especially compared to the naive



(a) $d = 20$ (b) $d = 200$

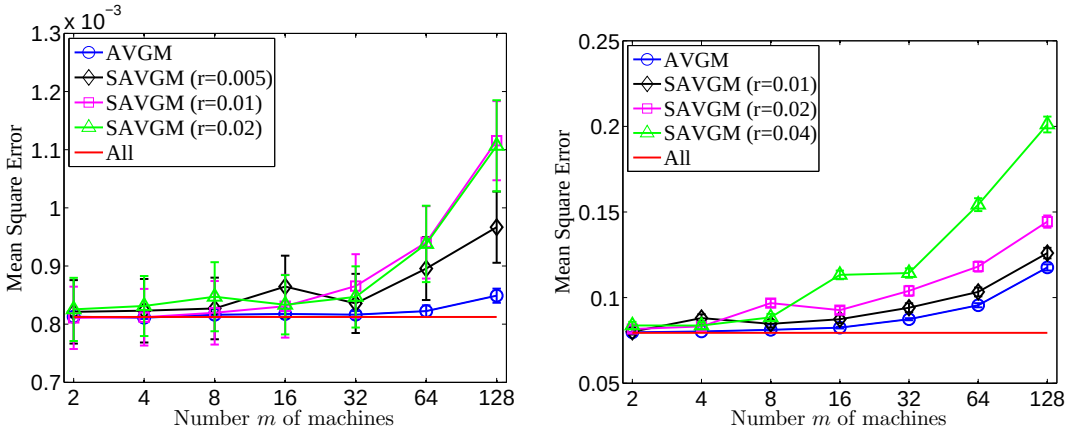
Figure 2: Comparison of AVGM and SGDAVGM methods as in Figure 1 plotted on logarithmic scale. The plot shows $\|\widehat{\theta} - \theta^*\|_2^2 - \|\theta_N - \theta^*\|_2^2$, where θ_N is the oracle least-squares estimator using all N data samples.

solution using only a fraction $1/m$ of the data. In particular, if $\widehat{\theta}$ is obtained by the batch method, then AVGM is almost as good as the full-batch baseline even for m as large as 128, though there is some evident degradation in solution quality. The SGDAVGM (stochastic-gradient with averaging) solution also yields much higher accuracy than the naive solution, but its performance degrades more quickly than the AVGM method's as m grows. In higher dimensions, both the AVGM and SGDAVGM procedures have somewhat worse performance; again, this is not unexpected since in high dimensions the strong convexity condition is satisfied with lower probability in local data sets.

We present a comparison between the AVGM method and the SGDAVGM method with somewhat more distinguishing power in Figure 2. For these plots, we compute the gap between the AVGM mean-squared-error and the unparallel baseline MSE, which is the accuracy lost due to parallelization or distributing the inference procedure across multiple machines. Figure 2 shows that the mean-squared error grows polynomially with the number of machines m, which is consistent with our theoretical results. From Corollary 3, we expect the AVGM method to suffer (lower-order) penalties proportional to m^2 as m grows, while Theorem 5 suggests the somewhat faster growth we see for the SGDAVGM method in Figure 2. Thus, we see that the improved run-time performance of the SGDAVGM method—requiring only a single pass through the data on each machine, touching each datum only once—comes at the expense of some loss of accuracy, as measured by mean-squared error.

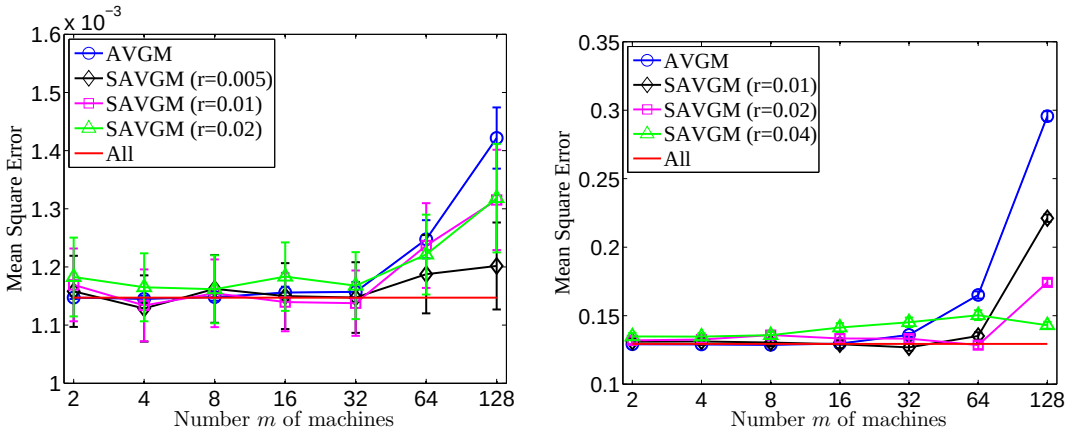
4.2 Subsampling Correction

We now turn to developing an understanding of the SAVGM algorithm in comparison to the standard average mixture algorithm, developing intuition for the benefits and drawbacks of the method. Before describing the results, we remark that for the standard regression model (16), the least-squares solution is unbiased for θ^*, so we expect subsampled averaging to yield little (if any) improvement. The SAVGM method is essentially aimed at correcting the bias of the estimator $\overline{\theta}_1$, and de-biasing an unbiased estimator only increases its variance. However, for the mis-specified models (17) and (18)



(a) $d = 20$ (b) $d = 200$

Figure 3: The error $\|\widehat{\theta} - \theta^*\|_2^2$ plotted against the number of machines m for the AVGM and SAVGM methods, with standard errors across twenty simulations, using the normal regression model (16). The oracle estimator is denoted by the line "All."

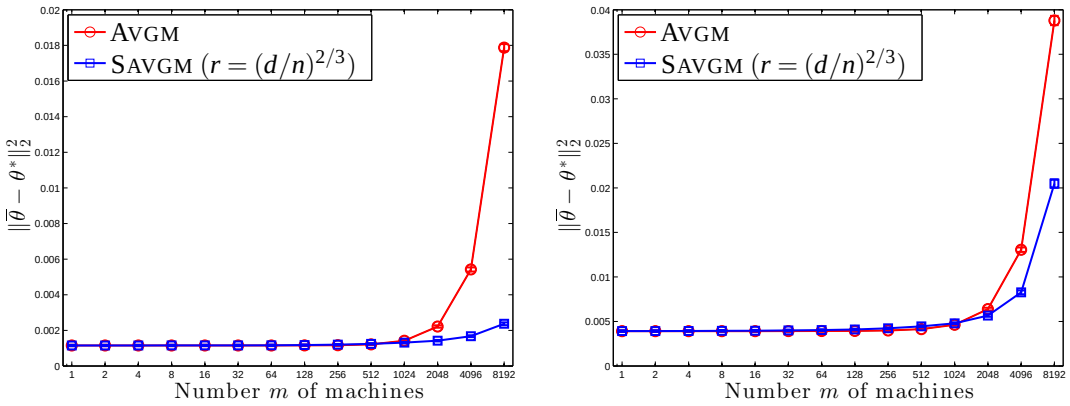


(a) $d = 20$ (b) $d = 200$

Figure 4: The error $\|\widehat{\theta} - \theta^*\|_2^2$ plotted against the number of machines m for the AVGM and SAVGM methods, with standard errors across twenty simulations, using the non-normal regression model (18). The oracle estimator is denoted by the line "All."

we expect to see some performance gains. In our experiments, we use multiple sub-sampling rates to study their effects, choosing $r \in \{0.005, 0.01, 0.02, 0.04\}$, where we recall that the output of the SAVGM algorithm is the vector $\widehat{\theta} := (\overline{\theta}_1 - r\overline{\theta}_2)/(1-r)$.

We begin with experiments in which the data is generated as in the previous section. That is, to generate a feature vector $x \in^d$, choose five distinct indices in $\{1, \ldots, d\}$ uniformly at random, and the entries of x at those indices are distributed as $N(0,1)$. In Figure 3, we plot the results of simulations comparing AVGM and SAVGM with data generated from the normal regression model (16). Both algorithms have have low error rates, but the AVGM method is slightly better than the SAVGM method for both values of the dimension d and all and sub-sampling rates r. As expected, in this



(a) $d = 20$ (b) $d = 40$

Figure 5: The error $\|\widehat{\theta} - \theta^*\|_2^2$ plotted against the number of machines m for the AVGM and SAVGM methods using regression model (17).

case the SAVGM method does not offer improvement over AVGM, since the estimators are unbiased. (In Figure 3(a), we note that the standard error is in fact very small, since the mean-squared error is only of order 10^{-3}.)

To understand settings in which subsampling for bias correction helps, in Figure 4, we plot mean-square error curves for the least-squares regression problem when the vector y is sampled according to the non-normal regression model (18). In this case, the least-squares estimator is biased for θ^* (which, as before, we estimate by solving a larger regression problem using $10N$ data samples). Figure 4 shows that both the AVGM and SAVGM method still enjoy good performance; in some cases, the SAVGM method even beats the oracle least-squares estimator for θ^* that uses all N samples. Since the AVGM estimate is biased in this case, its error curve increases roughly quadratically with m, which agrees with our theoretical predictions in Theorem 1. In contrast, we see that the SAVGM algorithm enjoys somewhat more stable performance, with increasing benefit as the number of machines m increases. For example, in case of $d = 200$, if we choose $r = 0.01$ for $m \leq 32$, choose $r = 0.02$ for $m = 64$ and $r = 0.04$ for $m = 128$, then SAVGM has performance comparable with the oracle method that uses all N samples. Moreover, we see that all the values of r—at least for the reasonably small values we use in the experiment—provide performance improvements over a non-subsampled distributed estimator.

For our final simulation, we plot results comparing SAVGM with AVGM in model (17), which is mis-specified but still a normal model. We use a simpler data generating mechanism, specifically, we draw $x \sim N(0, I_{d\times d})$ from a standard d-dimensional normal, and v is chosen uniformly in $[0,1]$; in this case, the population minimizer has the closed form $\theta^* = u + 3v$. Figure 5 shows the results for dimensions $d = 20$ and $d = 40$ performed over 100 experiments (the standard errors are too small to see). Since the model (17) is not that badly mis-specified, the performance of the SAVGM method improves upon that of the AVGM method only for relatively large values of m, however, the performance of the SAVGM is always at least as good as that of AVGM.

Feature Name	Dimension	Description
Query	20000	Word tokens appearing in the query.
Gender	3	Gender of the user
Keyword	20000	Word tokens appearing in the purchase keywords.
Title	20000	Word tokens appearing in the ad title.
Advertiser	39191	Advertiser's ID
AdID	641707	Advertisement's ID.
Age	6	Age of the user
UserFreq	25	Number of appearances of the same user.
Position	3	Position of advertisement on search page.
Depth	3	Number of ads in the session.
QueryFreq	25	Number of occurrences of the same query.
AdFreq	25	Number of occurrences of the same ad.
QueryLength	20	Number of words in the query.
TitleLength	30	Number of words in the ad title.
DespLength	50	Number of words in the ad description.
QueryCtr	150	Average click-through-rate for query.
UserCtr	150	Average click-through-rate for user.
AdvrCtr	150	Average click-through-rate for advertiser.
WordCtr	150	Average click-through-rate for keyword advertised.
UserAdFreq	20	Number of times this user sees an ad.
UserQueryFreq	20	Number of times this user performs a search.

Table 1: Features used in online advertisement prediction problem.

5. Experiments with Advertising Data

Predicting whether a user of a search engine will click on an advertisement presented to him or her is of central importance to the business of several internet companies, and in this section, we present experiments studying the performance of the AVGM and SAVGM methods for this task. We use a large data set from the Tencent search engine, `soso.com` (Sun, 2012), which contains 641,707 distinct advertisement items with $N = 235{,}582{,}879$ data samples.

Each sample consists of a so-called *impression*, which in the terminology of the information retrieval literature (e.g., see the book by Manning et al., 2008), is a list containing a user-issued search, the advertisement presented to the user in response to the search, and a label $y \in \{+1, -1\}$ indicating whether the user clicked on the advertisement. The ads in our data set were presented to 23,669,283 distinct users.

Transforming an impression into a useable set of regressors x is non-trivial, but the Tencent data set provides a standard encoding. We list the features present in the data in Table 1, along with some description of their meaning. Each text-based feature—that is, those made up of words, which are Query, Keyword, and Title—is given a "bag-of-words" encoding (Manning et al., 2008). This encoding assigns each of 20,000 possible words an index, and if the word appears in the query (or Keyword or Title feature), the corresponding index in the vector x is set to 1. Words that do not appear are encoded with a zero. Real-valued features, corresponding to the bottom fifteen features in Table 1 beginning with "Age", are binned into a fixed number of intervals $[-\infty, a_1], (a_1, a_2], \ldots, (a_k, \infty]$,

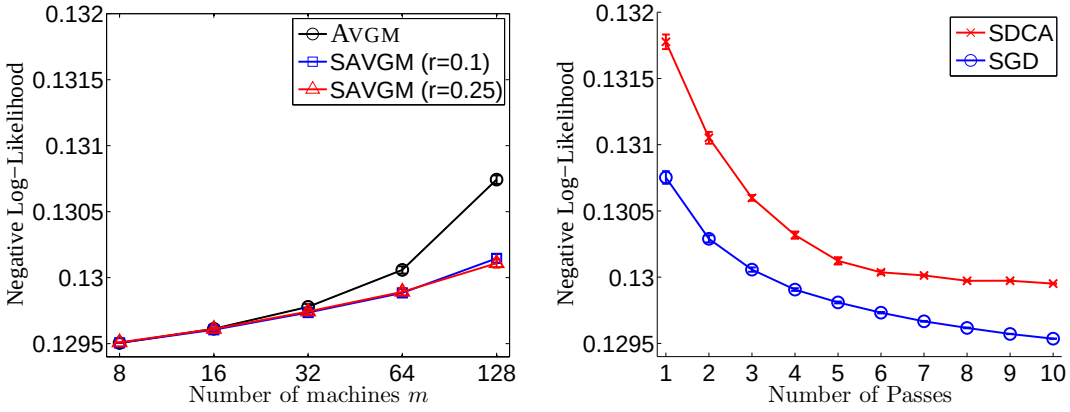


Figure 6: The negative log-likelihood of the output of the AVGM, SAVGM, and stochastic methods on the held-out data set for the click-through prediction task. (a) Performance of the AVGM and SAVGM methods versus the number of splits m of the data. (b) Performance of SDCA and SGD baselines as a function of number of passes through the entire data set.

each of which is assigned an index in x. (Note that the intervals and number thereof vary per feature, and the dimension of the features listed in Table 1 corresponds to the number of intervals). When a feature falls into a particular bin, the corresponding entry of x is assigned a 1, and otherwise the entries of x corresponding to the feature are 0. Each feature has one additional value for "unknown." The remaining categorical features—gender, advertiser, and advertisement ID (AdID)—are also given $\{0,1\}$ encodings, where only one index of x corresponding to the feature may be non-zero (which indicates the particular gender, advertiser, or AdID). This combination of encodings yields a binary-valued covariate vector $x \in \{0,1\}^d$ with $d = 741{,}725$ dimensions. Note also that the features incorporate information about the user, advertisement, and query issued, encoding information about their interactions into the model.

Our goal is to predict the probability of a user clicking a given advertisement as a function of the covariates in Table 1. To do so, we use a logistic regression model to estimate the probability of a click response

$$P(y = 1 \mid x; \theta) := \frac{1}{1 + \exp(-\langle \theta, x \rangle)},$$

where $\theta \in \mathbb{R}^d$ is the unknown regression vector. We use the negative logarithm of P as the loss, incorporating a ridge regularization penalty. This combination yields instantaneous loss

$$f(\theta; (x,y)) = \log\left(1 + \exp(-y \langle \theta, x \rangle)\right) + \frac{\lambda}{2} \|\theta\|_2^2.$$

In all our experiments, we assume that the population negative log-likelihood risk has local strong convexity as suggested by Assumption 2. In practice, we use a small regularization parameter $\lambda = 10^{-6}$ to ensure fast convergence for the local sub-problems.

For this problem, we cannot evaluate the mean-squared error $\|\widehat{\theta} - \theta^*\|_2^2$, as we do not know the true optimal parameter θ^*. Consequently, we evaluate the performance of an estimate $\widehat{\theta}$ using

log-loss on a held-out data set. Specifically, we perform a five-fold validation experiment, where we shuffle the data and partition it into five equal-sized subsets. For each of our five experiments, we hold out one partition to use as the test set, using the remaining data as the training set for inference. When studying the AVGM or SAVGM method, we compute the local estimate θ_i via a trust-region Newton-based method (Nocedal and Wright, 2006) implemented by LIBSVM (Chang and Lin, 2011).

The data set is too large to fit in the memory of most computers: in total, four splits of the data require 55 gigabytes. Consequently, it is difficult to provide an oracle training comparison using the full N samples. Instead, for each experiment, we perform 10 passes of stochastic dual coordinate ascent (SDCA) (Shalev-Shwartz and Zhang, 2012) and 10 passes of stochastic gradient descent (SGD) through the data set to get two rough baselines of the performance attained by the empirical minimizer for the entire training data set. Figure 6(b) shows the hold-out set log-loss after each of the sequential passes through the training data finishes. Note that although the SDCA enjoys faster convergence rate on the regularized empirical risk (Shalev-Shwartz and Zhang, 2012), the plot shows that the SGD has better generalization performance.

In Figure 6(a), we show the average hold-out set log-loss (with standard errors) of the estimator $\overline{\theta}_1$ provided by the AVGM method versus number of splits of the data m, and we also plot the log-loss of the SAVGM method using subsampling ratios of $r \in \{.1, .25\}$. The plot shows that for small m, both AVGM and SAVGM enjoy good performance, comparable to or better than (our proxy for) the oracle solution using all N samples. As the number of machines m grows, however, the de-biasing provided by the subsampled bootstrap method yields substantial improvements over the standard AVGM method. In addition, even with $m = 128$ splits of the data set, the SAVGM method gives better hold-out set performance than performing two passes of stochastic gradient on the entire data set of m samples; with $m = 64$, SAVGM enjoys performance as strong as looping through the data four times with stochastic gradient descent. This is striking, since doing even one pass through the data with stochastic gradient descent gives minimax optimal convergence rates (Polyak and Juditsky, 1992; Agarwal et al., 2012). In ranking applications, rather than measuring negative log-likelihood, one may wish to use a direct measure of prediction error; to that end, Figure 7 shows plots of the area-under-the-curve (AUC) measure for the AVGM and SAVGM methods; AUC is a well-known measure of prediction error for bipartite ranking (Manning et al., 2008). Broadly, this plot shows a similar story to that in Figure 6.

It is instructive and important to understand the sensitivity of the SAVGM method to the value of the resampling parameter r. We explore this question in Figure 8 using $m = 128$ splits, where we plot the log-loss of the SAVGM estimator on the held-out data set versus the subsampling ratio r. We choose $m = 128$ because more data splits provide more variable performance in r. For the `soso.com` ad prediction data set, the choice $r = .25$ achieves the best performance, but Figure 8 suggests that mis-specifying the ratio is not terribly detrimental. Indeed, while the performance of SAVGM degrades to that of the AVGM method, a wide range of settings of r give improved performance, and there does not appear to be a phase transition to poor performance.

6. Discussion

Large scale statistical inference problems are challenging, and the difficulty of solving them will only grow as data becomes more abundant: the amount of data we collect is growing much faster than the speed or storage capabilities of our computers. Our AVGM, SAVGM, and SGDAVGM meth-

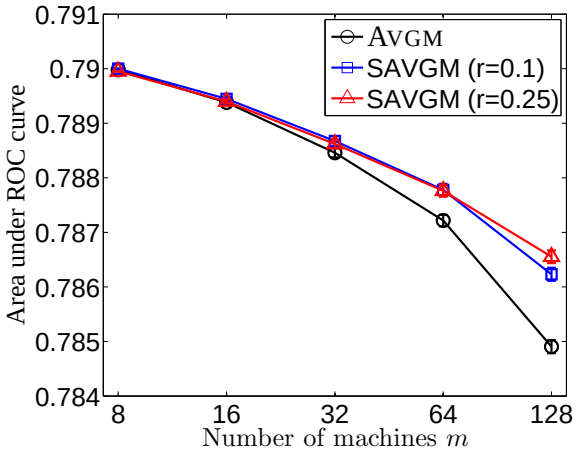


Figure 7: The area-under-the-curve (AUC) measure of ranking error for the output of the AVGM and SAVGM methods for the click-through prediction task.

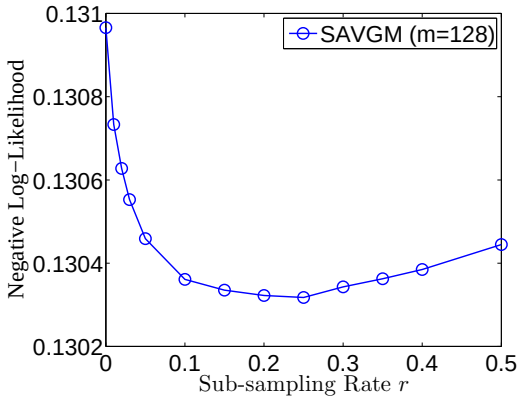


Figure 8: The log-loss on held-out data for the SAVGM method applied with $m = 128$ parallel splits of the data, plotted versus the sub-sampling rate r.

ods provide strategies for efficiently solving such large-scale risk minimization problems, enjoying performance comparable to an oracle method that is able to access the entire large data set. We believe there are several interesting questions that remain open after this work. First, nonparametric estimation problems, which often suffer superlinear scaling in the size of the data, may provide an interesting avenue for further study of decomposition-based methods. Our own recent work has addressed aspects of this challenge in the context of kernel methods for non-parametric regression (Zhang et al., 2013). More generally, an understanding of the interplay between statistical efficiency and communication could provide an avenue for further research, and it may also be interesting to study the effects of subsampled or bootstrap-based estimators in other distributed environments.

Acknowledgments

We thank Joel Tropp for some informative discussions on and references for matrix concentration and moment inequalities. We also thank Ohad Shamir for pointing out a mistake in the statements of results related to Theorem 1, and the editor and reviewers for their helpful comments and feedback. JCD was supported by the Department of Defence under the NDSEG Fellowship Program and by a Facebook PhD fellowship. This work was partially funded by Office of Naval Research MURI grant N00014-11-1-0688 to MJW.

Appendix A. The Necessity of Smoothness

Here we show that some version of the smoothness conditions presented in Assumption 3 are necessary for averaging methods to attain better mean-squared error than using only the n samples on a single processor. Given the loss function (5), let $n_0 = \sum_{i=1}^n 1_{(X_i=0)}$ to be the count of 0 samples. Using θ_1 as shorthand for $\theta_{1,i}$, we see by inspection that the empirical minimizer θ_1 is

$$\theta_1 = \begin{cases} \frac{n_0}{n} - \frac{1}{2} & \text{when } n_0 \le n/2 \\ 1 - \frac{n}{2n_0} & \text{otherwise.} \end{cases}$$

For simplicity, we may assume that n is odd. In this case, we obtain that

$$\begin{aligned} \mathbb{E}[\theta_1] &= \frac{1}{4} + \mathbb{E}\left[\frac{n_0}{n} 1_{(n_0 < n/2)}\right] - \mathbb{E}\left[\frac{n}{2n_0} 1_{(n_0 > n/2)}\right] \\ &= \frac{1}{4} + \frac{1}{2^n} \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{i} \frac{i}{n} - \frac{1}{2^n} \sum_{i=\lceil n/2 \rceil}^{n} \binom{n}{i} \frac{n}{2i} = \frac{1}{4} + \frac{1}{2^n} \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{i} \left[\frac{i}{n} - \frac{n}{2(n-i)}\right] \end{aligned}$$

by the symmetry of the binomial. Adding and subtracting $\frac{1}{2}$ from the term within the braces, noting that $P(n_0 < n/2) = 1/2$, we have the equality

$$\mathbb{E}[\theta_1] = \frac{1}{2^n} \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{i} \left[\frac{i}{n} - \frac{n}{2(n-i)} + \frac{1}{2}\right] = \frac{1}{2^n} \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{i} \frac{i(n-2i)}{2n(n-i)}.$$

If Z is distributed normally with mean $1/2$ and variance $1/(4n)$, then an asymptotic expansion of the binomial distribution yields

$$\begin{aligned} \left(\frac{1}{2}\right)^n \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{i} \frac{i(n-2i)}{2n(n-i)} &= \mathbb{E}\left[\frac{Z(1-2Z)}{2-2Z} \mid 0 \le Z \le \frac{1}{2}\right] + o(n^{-1/2}) \\ &\ge \frac{1}{2}\mathbb{E}\left[Z - 2Z^2 \mid 0 \le Z \le \frac{1}{2}\right] + o(n^{-1/2}) = \Omega(n^{-\frac{1}{2}}), \end{aligned}$$

the final equality following from standard calculations, since $\mathbb{E}[|Z|] = \Omega(n^{-1/2})$.

Appendix B. Proof of Theorem 1

Although Theorem 1 is in terms of bounds on 8^{th} order moments, we prove a somewhat more general result in terms of a set of (k_0, k_1, k_2) moment conditions given by

$$\mathbb{E}[\|\nabla f(\theta; X)\|_2^{k_0}] \le G^{k_0}, \qquad \mathbb{E}[\left|\left|\left|\nabla^2 f(\theta; X) - \nabla^2 F_0(\theta)\right|\right|\right|_2^{k_1}] \le H^{k_1},$$
$$\mathbb{E}[L(X)^{k_2}] \le L^{k_2} \quad \text{and} \quad \mathbb{E}[(L(X) - \mathbb{E}[L(X)])^{k_2}] \le L^{k_2}$$

for $\theta \in U$. (Recall the definition of U prior to Assumption 3). Doing so allows sharper control if higher moment bounds are available. The reader should recall throughout our arguments that we have assumed $\min\{k_0, k_1, k_2\} \ge 8$. Throughout the proof, we use F_1 and θ_1 to indicate the local empirical objective and empirical minimizer of machine 1 (which have the same distribution as those of the other processors), and we recall the notation $1_{(\mathcal{E})}$ for the indicator function of the event $\mathcal{E}$.

Before beginning the proof of Theorem 1 proper, we begin with a simple inequality that relates the error term $\overline{\theta} - \theta^*$ to an average of the errors $\theta_i - \theta^*$, each of which we can bound in turn. Specifically, a bit of algebra gives us that

$$\begin{aligned}
\mathbb{E}[\left\|\overline{\theta} - \theta^*\right\|_2^2] &= \mathbb{E}\left[\left\|\frac{1}{m}\sum_{i=1}^m \theta_i - \theta^*\right\|_2^2\right] \\
&= \frac{1}{m^2}\sum_{i=1}^m \mathbb{E}[\|\theta_i - \theta^*\|_2^2] + \frac{1}{m^2}\sum_{i \ne j} \mathbb{E}[\left\langle \theta_i - \theta^*, \theta_j - \theta^* \right\rangle] \\
&\le \frac{1}{m}\mathbb{E}[\|\theta_1 - \theta^*\|_2^2] + \frac{m(m-1)}{m^2}\left\|\mathbb{E}[\theta_1 - \theta^*]\right\|_2^2 \\
&\le \frac{1}{m}\mathbb{E}[\|\theta_1 - \theta^*\|_2^2] + \left\|\mathbb{E}[\theta_1 - \theta^*]\right\|_2^2. \qquad (19)
\end{aligned}$$

Here we used the definition of the averaged vector $\overline{\theta}$ and the fact that for $i \ne j$, the vectors θ_i and θ_j are statistically independent, they are functions of independent samples. The upper bound (19) illuminates the path for the remainder of our proof: we bound each of $\mathbb{E}[\|\theta_i - \theta^*\|_2^2]$ and $\|\mathbb{E}[\theta_i - \theta^*]\|_2^2$. Intuitively, since our objective is locally strongly convex by Assumption 2, the empirical minimizing vector θ_1 is a nearly unbiased estimator for θ^*, which allows us to prove the convergence rates in the theorem.

We begin by defining three events—which we (later) show hold with high probability—that guarantee the closeness of θ_1 and θ^*. In rough terms, when these events hold, the function F_1 behaves similarly to the population risk F_0 around the point θ^*; since F_0 is locally strongly convex, the minimizer θ_1 of F_1 will be close to θ^*. Recall that Assumption 3 guarantees the existence of a ball $U_\rho = \{\theta \in \mathbb{R}^d : \|\theta - \theta^*\|_2 < \rho\}$ of radius $\rho \in (0,1)$ such that

$$\left|\left|\left|\nabla^2 f(\theta; x) - \nabla^2 f(\theta'; x)\right|\right|\right|_2 \le L(x)\left\|\theta - \theta'\right\|_2$$

for all $\theta, \theta' \in U_\rho$ and any x, where $\mathbb{E}[L(X)^{k_2}] \le L^{k_2}$. In addition, Assumption 2 guarantees that $\nabla^2 F_0(\theta^*) \succeq \lambda I$. Now, choosing the potentially smaller radius $\delta_\rho = \min\{\rho, \rho\lambda/4L\}$, we can define

the three "good" events

$$\mathcal{E}_0 := \left\{ \frac{1}{n} \sum_{i=1}^{n} L(X_i) \leq 2L \right\},$$
$$\mathcal{E}_1 := \left\{ \left| \left| \left| \nabla^2 F_1(\theta^*) - \nabla^2 F_0(\theta^*) \right| \right| \right|_2 \leq \frac{\rho\lambda}{2} \right\}, \quad \text{and} \tag{20}$$
$$\mathcal{E}_2 := \left\{ \|\nabla F_1(\theta^*)\|_2 \leq \frac{(1-\rho)\lambda\delta_\rho}{2} \right\}.$$

We then have the following lemma:

Lemma 6 *Under the events $\mathcal{E}_0$, $\mathcal{E}_1$, and $\mathcal{E}_2$ previously defined* (20)*, we have*

$$\|\theta_1 - \theta^*\|_2 \leq \frac{2\|\nabla F_1(\theta^*)\|_2}{(1-\rho)\lambda}, \quad \textit{and} \quad \nabla^2 F_1(\theta) \succeq (1-\rho)\lambda I_{d\times d}.$$

The proof of Lemma 6 relies on some standard optimization guarantees relating gradients to minimizers of functions (e.g., Boyd and Vandenberghe, 2004, Chapter 9), although some care is required since smoothness and strong convexity hold only locally in our problem. As the argument is somewhat technical, we defer it to Appendix E.

Our approach from here is to give bounds on $\mathbb{E}[\|\theta_1 - \theta^*\|_2^2]$ and $\|\mathbb{E}[\theta_1 - \theta^*]\|_2^2$ by careful Taylor expansions, which allows us to bound $\mathbb{E}[\|\overline{\theta}_1 - \theta^*\|_2^2]$ via our initial expansion (19). We begin by noting that whenever the events $\mathcal{E}_0$, $\mathcal{E}_1$, and $\mathcal{E}_2$ hold, then $\nabla F_1(\theta_1) = 0$, and moreover, by a Taylor series expansion of ∇F_1 between θ^* and θ_1, we have

$$0 = \nabla F_1(\theta_1) = \nabla F_1(\theta^*) + \nabla^2 F_1(\theta')(\theta_1 - \theta^*)$$

where $\theta' = \kappa\theta^* + (1-\kappa)\theta_1$ for some $\kappa \in [0,1]$. By adding and subtracting terms, we have

$$0 = \nabla F_1(\theta^*) + (\nabla^2 F_1(\theta') - \nabla^2 F_1(\theta^*))(\theta_1 - \theta^*) + (\nabla^2 F_1(\theta^*) - \nabla^2 F_0(\theta^*))(\theta_1 - \theta^*) + \nabla^2 F_0(\theta^*)(\theta_1 - \theta^*). \tag{21}$$

Since $\nabla^2 F_0(\theta^*) \succeq \lambda I$, we can define the inverse Hessian matrix $\Sigma^{-1} := [\nabla^2 F_0(\theta^*)]^{-1}$, and setting $\Delta := \theta_1 - \theta^*$, we multiply both sides of the Taylor expansion (21) by Σ^{-1} to obtain the relation

$$\Delta = -\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(\nabla^2 F_1(\theta^*) - \nabla^2 F_1(\theta'))\Delta + \Sigma^{-1}(\nabla^2 F_0(\theta^*) - \nabla^2 F_1(\theta^*))\Delta. \tag{22}$$

Thus, if we define the matrices $P = \nabla^2 F_0(\theta^*) - \nabla^2 F_1(\theta^*)$ and $Q = \nabla^2 F_1(\theta^*) - \nabla^2 F_1(\theta')$, equality (22) can be re-written as

$$\theta_1 - \theta^* = -\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)(\theta_1 - \theta^*). \tag{23}$$

Note that Equation (23) holds when the conditions of Lemma 6 hold, and otherwise we may simply assert only that $\|\theta_1 - \theta^*\|_2 \leq R$. Roughly, we expect the final two terms in the error expansion (23) to be of smaller order than the first term, since we hope that $\theta_1 - \theta^* \to 0$ and additionally that the Hessian differences decrease to zero at a sufficiently fast rate. We now formalize this intuition.

Inspecting the Taylor expansion (23), we see that there are several terms of a form similar to $(\nabla^2 F_0(\theta^*) - \nabla^2 F_1(\theta^*))(\theta_1 - \theta^*)$; using the smoothness Assumption 3, we can convert these terms into higher order terms involving only $\theta_1 - \theta^*$. Thus, to effectively control the expansions (22) and (23), we must show that higher order terms of the form $\mathbb{E}[\|\theta_1 - \theta^*\|_2^k]$, for $k \geq 2$, decrease quickly enough in n.

B.0.1 CONTROL OF $\mathbb{E}[\|\theta_1 - \theta^*\|_2^k]$

Recalling the events (20), we define $\mathcal{E} := \mathcal{E}_0 \cap \mathcal{E}_1 \cap \mathcal{E}_2$ and then observe that

$$\begin{aligned}\mathbb{E}[\|\theta_1 - \theta^*\|_2^k] &= \mathbb{E}[1_{(\mathcal{E})} \|\theta_1 - \theta^*\|_2^k] + \mathbb{E}[1_{(\mathcal{E}^c)} \|\theta_1 - \theta^*\|_2^k] \\ &\leq \frac{2^k \mathbb{E}[1_{(\mathcal{E})} \|\nabla F_1(\theta^*)\|_2^k]}{(1-\rho)^k \lambda^k} + \mathbb{P}(\mathcal{E}^c) R^k \\ &\leq \frac{2^k \mathbb{E}[\|\nabla F_1(\theta^*)\|_2^k]}{(1-\rho)^k \lambda^k} + \mathbb{P}(\mathcal{E}^c) R^k,\end{aligned}$$

where we have used the bound $\|\theta - \theta^*\|_2 \leq R$ for all $\theta \in \Theta$, from Assumption 1. Our goal is to prove that $\mathbb{E}[\|\nabla F_1(\theta^*)\|_2^k] = O(n^{-k/2})$ and that $\mathbb{P}(\mathcal{E}^c) = O(n^{-k/2})$. We move forward with a two lemmas that lay the groundwork for proving these two facts:

Lemma 7 *Under Assumption 3, there exist constants C and C' (dependent only on the moments k_0 and k_1 respectively) such that*

$$\mathbb{E}[\|\nabla F_1(\theta^*)\|_2^{k_0}] \leq C \frac{G^{k_0}}{n^{k_0/2}}, \quad \text{and} \tag{24}$$

$$\mathbb{E}[\left|\left|\left|\nabla^2 F_1(\theta^*) - \nabla^2 F_0(\theta^*)\right|\right|\right|_2^{k_1}] \leq C' \frac{\log^{k_1/2}(2d) H^{k_1}}{n^{k_1/2}}. \tag{25}$$

See Appendix F for the proof of this claim.

As an immediate consequence of Lemma 7, we see that the events $\mathcal{E}_1$ and $\mathcal{E}_2$ defined by (20) occur with reasonably high probability. Indeed, recalling that $\mathcal{E} = \mathcal{E}_0 \cap \mathcal{E}_1 \cap \mathcal{E}_2$, Boole's law and the union bound imply

$$\begin{aligned}&\mathbb{P}(\mathcal{E}^c) = \mathbb{P}(\mathcal{E}_0^c \cup \mathcal{E}_1^c \cup \mathcal{E}_2^c) \\ &\leq \mathbb{P}(\mathcal{E}_0^c) + \mathbb{P}(\mathcal{E}_1^c) + \mathbb{P}(\mathcal{E}_2^c) \\ &\leq \frac{\mathbb{E}[|\frac{1}{n}\sum_{i=1}^n L(X_i) - \mathbb{E}[L(X)]|^{k_2}]}{L^{k_2}} + \frac{2^{k_1}\mathbb{E}[\left|\left|\left|\nabla^2 F_1(\theta^*) - \nabla^2 F_0(\theta^*)\right|\right|\right|_2^{k_1}]}{\rho^{k_1}\lambda^{k_1}} + \frac{2^{k_0}\mathbb{E}[\|\nabla F_1(\theta^*)\|_2^{k_0}]}{(1-\rho)^{k_0}\lambda^{k_0}\delta_\rho^{k_0}} \\ &\leq C_2 \frac{1}{n^{k_2/2}} + C_1 \frac{\log^{k_1/2}(2d) H^{k_1}}{n^{k_1/2}} + C_0 \frac{G^{k_0}}{n^{k_0/2}}\end{aligned} \tag{26}$$

for some universal constants C_0, C_1, C_2, where in the second-to-last line we have invoked the moment bound in Assumption 3. Consequently, we find that

$$\mathbb{P}(\mathcal{E}^c) R^k = O(R^k(n^{-k_1/2} + n^{-k_2/2} + n^{-k_0/2}) \quad \text{for any } k \in \mathbb{N}.$$

In summary, we have proved the following lemma:

Lemma 8 *Let Assumptions 2 and 3 hold. For any $k \in \mathbb{N}$ with $k \leq \min\{k_0, k_1, k_2\}$, we have*

$$\mathbb{E}[\|\theta_1 - \theta^*\|_2^k] = O\left(n^{-k/2} \cdot \frac{G^k}{(1-\rho)^k \lambda^k} + n^{-k_0/2} + n^{-k_1/2} + n^{-k_2/2}\right) = O\left(n^{-k/2}\right),$$

where the order statements hold as $n \to +\infty$.

Now recall the matrix $Q = \nabla^2 F_1(\theta^*) - \nabla^2 F_1(\theta')$ defined following Equation (22). The following result controls the moments of its operator norm:

Lemma 9 *For $k \leq \min\{k_2, k_1, k_0\}/2$, we have $\mathbb{E}[|||Q|||_2^k] = O(n^{-k/2})$.*

Proof We begin by using Jensen's inequality and Assumption 3 to see that

$$|||Q|||^k \leq \frac{1}{n}\sum_{i=1}^n \left|\left|\left|\nabla^2 f(\theta'; X_i) - \nabla^2 f(\theta^*; X_i)\right|\right|\right|^k \leq \frac{1}{n}\sum_{i=1}^n L(X_i)^k \left\|\theta' - \theta^*\right\|_2^k.$$

Now we apply the Cauchy-Schwarz inequality and Lemma 8, thereby obtaining

$$\mathbb{E}[|||Q|||_2^k] \leq \mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n L(X_i)^k\right)^2\right]^{\frac{1}{2}} \mathbb{E}\left[\|\theta_1 - \theta^*\|_2^{2k}\right]^{\frac{1}{2}} = O\left(L^k \frac{G^k}{(1-\rho)^k \lambda^k} n^{-k/2}\right),$$

where we have used Assumption 3 again. ■

Lemma 8 allows us to control the first term from our initial bound (19) almost immediately. Indeed, using our last Taylor expansion (23) and the definition of the event $\mathcal{E} = \mathcal{E}_0 \cap \mathcal{E}_1 \cap \mathcal{E}_2$, we have

$$\begin{aligned}\mathbb{E}[\|\theta_1 - \theta^*\|_2^2] &= \mathbb{E}\left[1_{(\mathcal{E})}\left\|-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)(\theta_1 - \theta^*)\right\|_2^2\right] + \mathbb{E}[1_{(\mathcal{E}^c)}\|\theta_1 - \theta^*\|_2^2] \\ &\leq 2\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right] + 2\mathbb{E}\left[\left\|\Sigma^{-1}(P+Q)(\theta_1 - \theta^*)\right\|_2^2\right] + \mathbb{P}(\mathcal{E}^c)R^2,\end{aligned}$$

where we have applied the inequality $(a+b)^2 \leq 2a^2 + 2b^2$. Again using this same inequality, then applying Cauchy-Schwarz and Lemmas 8 and 9, we see that

$$\begin{aligned}\mathbb{E}\left[\left\|\Sigma^{-1}(P+Q)(\theta_1 - \theta^*)\right\|_2^2\right] &\leq 2\left|\left|\left|\Sigma^{-1}\right|\right|\right|_2^2 \left(\mathbb{E}[|||P|||_2^2 \|\theta_1 - \theta^*\|_2^2] + \mathbb{E}[|||Q|||_2^2 \|\theta_1 - \theta^*\|_2^2]\right) \\ &\leq 2\left|\left|\left|\Sigma^{-1}\right|\right|\right|_2^2 \left(\sqrt{\mathbb{E}[|||P|||_2^4]\mathbb{E}[\|\theta_1 - \theta^*\|_2^4]} + \sqrt{\mathbb{E}[|||Q|||_2^4]\mathbb{E}[\|\theta_1 - \theta^*\|_2^4]}\right) \\ &= O(n^{-2}),\end{aligned}$$

where we have used the fact that $\min\{k_0, k_1, k_2\} \geq 8$ to apply Lemma 9. Combining these results, we obtain the upper bound

$$\mathbb{E}[\|\theta_1 - \theta^*\|_2^2] \leq 2\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right] + O(n^{-2}), \tag{27}$$

which completes the first part of our proof of Theorem 1.

B.0.2 CONTROL OF $\|\mathbb{E}[\theta_1 - \theta^*]\|_2^2$

It remains to consider the $\|\mathbb{E}[\theta_1 - \theta^*]\|_2^2$ term from our initial error inequality (19). When the events (20) occur, we know that all derivatives exist, so we may recursively apply our expansion (23) of $\theta_1 - \theta^*$ to find that

$$\begin{aligned}\theta_1 - \theta^* &= -\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)(\theta_1 - \theta^*) \\ &= \underbrace{-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)\left[-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)(\theta_1 - \theta^*)\right]}_{=:v}\end{aligned} \tag{28}$$

where we have introduced v as shorthand for the vector on the right hand side. Thus, with a bit of algebraic manipulation we obtain the relation

$$\theta_1 - \theta^* = 1_{(\mathcal{E})} v + 1_{(\mathcal{E}^c)}(\theta_1 - \theta^*) = v + 1_{(\mathcal{E}^c)}(\theta_1 - \theta^*) - 1_{(\mathcal{E}^c)} v = v + 1_{(\mathcal{E}^c)}(\theta_1 - \theta^* - v). \quad (29)$$

Now note that $\mathbb{E}[\nabla F_1(\theta^*)] = 0$ thus

$$\begin{aligned}\mathbb{E}[v] &= \mathbb{E}\left[-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)[-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)(\theta_1 - \theta^*)]\right] \\ &= \mathbb{E}\left[\Sigma^{-1}(P+Q)\Sigma^{-1}\left[(P+Q)(\theta_1 - \theta^*) - \nabla F_1(\theta^*)\right]\right].\end{aligned}$$

Thus, by re-substituting the appropriate quantities in (29) and applying the triangle inequality, we have

$$\begin{aligned}&\|\mathbb{E}[\theta_1 - \theta^*]\|_2 \\ &\leq \left\|\mathbb{E}[\Sigma^{-1}(P+Q)\Sigma^{-1}\left((P+Q)(\theta_1 - \theta^*) - \nabla F_1(\theta^*)\right)]\right\|_2 + \left\|\mathbb{E}[1_{(\mathcal{E}^c)}(\theta_1 - \theta^* - v)]\right\|_2 \\ &\leq \left\|\mathbb{E}[\Sigma^{-1}(P+Q)\Sigma^{-1}\left((P+Q)(\theta_1 - \theta^*) - \nabla F_1(\theta^*)\right)]\right\|_2 + \mathbb{E}[1_{(\mathcal{E}^c)}\|\theta_1 - \theta^*\|_2] \\ &\quad + \mathbb{E}\left[1_{(\mathcal{E}^c)}\left\|-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)\Sigma^{-1}\left[-\nabla F_1(\theta^*) + (P+Q)(\theta_1 - \theta^*)\right]\right\|_2\right]. \quad (30)\end{aligned}$$

Since $\|\theta_1 - \theta^*\|_2 \leq R$ by assumption, we have

$$\mathbb{E}[1_{(\mathcal{E}^c)}\|\theta_1 - \theta^*\|_2] \leq \mathbb{P}(\mathcal{E}^c)R \overset{(i)}{=} O(Rn^{-k/2})$$

for any $k \leq \min\{k_2, k_1, k_0\}$, where step (i) follows from the inequality (26). Hölder's inequality also yields that

$$\begin{aligned}\mathbb{E}\left[1_{(\mathcal{E}^c)}\left\|\Sigma^{-1}(P+Q)\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2\right] &\leq \mathbb{E}\left[1_{(\mathcal{E}^c)}\left|\left|\left|\Sigma^{-1}(P+Q)\right|\right|\right|_2\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2\right] \\ &\leq \sqrt{\mathbb{P}(\mathcal{E}^c)}\mathbb{E}\left[\left|\left|\left|\Sigma^{-1}(P+Q)\right|\right|\right|_2^4\right]^{1/4}\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^4\right]^{1/4}.\end{aligned}$$

Recalling Lemmas 7 and 9, we have $\mathbb{E}[\left|\left|\left|\Sigma^{-1}(P+Q)\right|\right|\right|_2^4] = O(\log^2(d)n^{-2})$, and we similarly have $\mathbb{E}[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^4] = O(n^{-2})$. Lastly, we have $\mathbb{P}(\mathcal{E}^c) = O(n^{-k/2})$ for $k \leq \min\{k_0, k_1, k_2\}$, whence we find that for any such k,

$$\mathbb{E}\left[1_{(\mathcal{E}^c)}\left\|\Sigma^{-1}(P+Q)\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2\right] = O\left(\sqrt{\log(d)}n^{-k/4-1}\right).$$

We can similarly apply Lemma 8 to the last remaining term in the inequality (30) to obtain that for any $k \leq \min\{k_2, k_1, k_0\}$,

$$\begin{aligned}&\mathbb{E}\left[1_{(\mathcal{E}^c)}\left\|-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)\left[-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(P+Q)(\theta_1 - \theta^*)\right]\right\|_2\right] \\ &\qquad = O(n^{-k/2} + n^{-k/4-1}).\end{aligned}$$

Applying these two bounds, we find that

$$\|\mathbb{E}[\theta_1 - \theta^*]\|_2 \leq \left\|\mathbb{E}\left[\Sigma^{-1}(P+Q)\Sigma^{-1}\left((P+Q)(\theta_1 - \theta^*) - \nabla F_1(\theta^*)\right)\right]\right\|_2 + O(n^{-k}) \quad (31)$$

for any k such that $k \leq \min\{k_0, k_1, k_2\}/2$ and $k \leq \min\{k_0, k_1, k_2\}/4 + 1$.

In the remainder of the proof, we show that part of the bound (31) still consists only of higher-order terms, leaving us with an expression not involving $\theta_1 - \theta^*$. To that end, note that

$$\mathbb{E}\left[\left\|\Sigma^{-1}(P+Q)\Sigma^{-1}(P+Q)(\theta_1-\theta^*)\right\|_2^2\right] = O(n^{-3})$$

by three applications of Hölder's inequality, the fact that $\|Ax\|_2 \leq |||A|||_2 \|x\|_2$, and Lemmas 7, 8 and 9. Coupled with our bound (31), we use the fact that $(a+b)^2 \leq 2a^2+2b^2$ to obtain

$$\|\mathbb{E}[\theta_1-\theta^*]\|_2^2 \leq 2\left\|\mathbb{E}[\Sigma^{-1}(P+Q)\Sigma^{-1}\nabla F_1(\theta^*)]\right\|_2^2 + O(n^{-3}). \tag{32}$$

We focus on bounding the remaining expectation. We have the following series of inequalities:

$$\begin{aligned}
\left\|\mathbb{E}[\Sigma^{-1}(P+Q)\Sigma^{-1}\nabla F_1(\theta^*)]\right\|_2 &\overset{(i)}{\leq} \mathbb{E}\left[|||\Sigma^{-1}(P+Q)|||_2 \left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2\right] \\
&\overset{(ii)}{\leq} \left(\mathbb{E}\left[|||\Sigma^{-1}(P+Q)|||_2^2\right]\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right]\right)^{\frac{1}{2}} \\
&\overset{(iii)}{\leq} \left(2\mathbb{E}\left[|||\Sigma^{-1}P|||_2^2 + |||\Sigma^{-1}Q|||_2^2\right]\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right]\right)^{\frac{1}{2}}.
\end{aligned}$$

Here step (i) follows from Jensen's inequality and the fact that $\|Ax\|_2 \leq |||A|||_2 \|x\|_2$; step (ii) uses the Cauchy-Schwarz inequality; and step (iii) follows from the fact that $(a+b)^2 \leq 2a^2+2b^2$. We have already bounded the first two terms in the product in our proofs; in particular, Lemma 7 guarantees that $\mathbb{E}[|||P|||_2^2] \leq CH\log d/n$, while

$$\mathbb{E}[|||Q|||_2^2] \leq \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^n L(X_i)^4\right]^{\frac{1}{2}} \mathbb{E}[\|\theta_1-\theta^*\|_2^4]^{\frac{1}{2}} \leq C\frac{L^2G^2}{(1-\rho)^2\lambda^2}\cdot n^{-1}$$

for some numerical constant C (recall Lemma 9). Summarizing our bounds on $|||P|||_2$ and $|||Q|||_2$, we have

$$\begin{aligned}
&\left\|\mathbb{E}\left[\Sigma^{-1}(P+Q)\Sigma^{-1}\nabla F_1(\theta^*)\right]\right\|_2^2 \\
&\leq 2\left|||\Sigma^{-1}|||_2^2\left(\frac{2H^2(\log d+1)}{n} + 2C\frac{L^2G^2}{(1-\rho)^2\lambda^2 n} + O(n^{-2})\right)\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right].
\end{aligned} \tag{33}$$

From Assumption 3 we know that $\mathbb{E}[\|\nabla F_1(\theta^*)\|_2^2] \leq G^2/n$ and $\left|||\Sigma^{-1}|||_2 \leq 1/\lambda$, and hence we can further simplify the bound (33) to obtain

$$\begin{aligned}
\|\mathbb{E}[\theta_1-\theta^*]\|_2^2 &\leq \frac{C}{\lambda^2}\left(\frac{H^2\log d + L^2G^2/\lambda^2(1-\rho)^2}{n}\right)\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right] + O(n^{-3}) \\
&= \frac{C}{\lambda^2}\left(\frac{H^2\log d + L^2G^2/\lambda^2(1-\rho)^2}{n^2}\right)\mathbb{E}\left[\left\|\Sigma^{-1}\nabla f(\theta^*;X)\right\|_2^2\right] + O(n^{-3})
\end{aligned}$$

for some numerical constant C, where we have applied our earlier inequality (32). Noting that we may (without loss of generality) take $\rho < \frac{1}{2}$, then applying this inequality with the bound (27) on $\mathbb{E}[\|\theta_1-\theta^*\|_2^2]$ we previously proved to our decomposition (19) completes the proof.

Appendix C. Proof of Theorem 4

Our proof of Theorem 4 begins with a simple inequality that mimics our first inequality (19) in the proof of Theorem 1. Recall the definitions of the averaged vector $\bar{\theta}_1$ and subsampled averaged vector $\bar{\theta}_2$. Let θ_1 denote the minimizer of the (an arbitrary) empirical risk F_1, and θ_2 denote the minimizer of the resampled empirical risk F_2 (from the same samples as θ_1). Then we have

$$\mathbb{E}\left[\left\|\frac{\bar{\theta}_1 - r\bar{\theta}_2}{1-r} - \theta^*\right\|_2^2\right] \le \left\|\mathbb{E}\left[\frac{\theta_1 - r\theta_2}{1-r} - \theta^*\right]\right\|_2^2 + \frac{1}{m}\mathbb{E}\left[\left\|\frac{\theta_1 - r\theta_2}{1-r} - \theta^*\right\|_2^2\right]. \tag{34}$$

Thus, parallel to our proof of Theorem 1, it suffices to bound the two terms in the decomposition (34) separately. Specifically, we prove the following two lemmas.

Lemma 10 *Under the conditions of Theorem 4,*

$$\left\|\mathbb{E}\left[\frac{\theta_1 - r\theta_2}{1-r} - \theta^*\right]\right\|_2^2 \le O(1)\frac{1}{r(1-r)^2}\left(\frac{M^2G^6}{\lambda^6} + \frac{G^4L^2}{\lambda^4}d\log d\right)\frac{1}{n^3}. \tag{35}$$

Lemma 11 *Under the conditions of Theorem 4,*

$$\mathbb{E}\left[\|\theta_1 - \theta^* - r(\theta_2 - \theta^*)\|_2^2\right] \le (2+3r)\mathbb{E}\left[\left\|\nabla^2 F_0(\theta^*)^{-1}\nabla F_1(\theta^*)\right\|_2^2\right] + O(n^{-2}) \tag{36}$$

In conjunction, Lemmas 10 and 11 coupled with the decomposition (34) yield the desired claim. Indeed, applying each of the lemmas to the decomposition (34), we see that

$$\begin{aligned}\mathbb{E}\left[\left\|\frac{\bar{\theta}_1 - r\bar{\theta}_2}{1-r} - \theta^*\right\|_2^2\right] &\le \frac{2+3r}{(1-r)^2 m}\mathbb{E}\left[\left\|\nabla^2 F_0(\theta^*)^{-1}\nabla F_1(\theta^*)\right\|_2^2\right] \\ &\quad + O\left(\frac{1}{(1-r)^2}m^{-1}n^{-2}\right) + O\left(\frac{1}{r(1-r)^2}n^{-3}\right),\end{aligned}$$

which is the statement of Theorem 4.

The remainder of our argument is devoted to establishing Lemmas 10 and 11. Before providing their proofs (in Appendices C.3 and C.4 respectively), we require some further set-up and auxiliary results. Throughout the rest of the proof, we use the notation

$$Y = Y' + \mathcal{R}_k$$

for some random variables Y and Y' to mean that there exists a random variable Z such that $Y = Y' + Z$ and $\mathbb{E}[\|Z\|_2^2] = O(n^{-k})$.[1] The symbol $\mathcal{R}_k$ may indicate different random variables throughout a proof and is notational shorthand for a moment-based big-O notation. We also remark that if we have $\mathbb{E}[\|Z\|_2^2] = O(a^k n^{-k})$, we have $Z = a^{k/2}\mathcal{R}_k$, since $(a^{k/2})^2 = a^k$. For shorthand, we also say that $\mathbb{E}[Z] = O(h(n))$ if $\|\mathbb{E}[Z]\|_2 = O(h(n))$, which implies that if $Z = \mathcal{R}_k$ then $\mathbb{E}[Z] = O(n^{-k/2})$, since

$$\|\mathbb{E}[Z]\|_2 \le \sqrt{\mathbb{E}[\|Z\|_2^2]} = O(n^{-k/2}).$$

1. Formally, in our proof this will mean that there exist random vectors Y, Y', and Z that are measurable with respect to the σ-field $\sigma(X_1, \ldots, X_n)$, where $Y = Y' + Z$ and $\mathbb{E}[\|Z\|_2^2] = O(n^{-k})$.

C.1 Optimization Error Expansion

In this section, we derive a sharper asymptotic expansion of the optimization errors $\theta_1 - \theta^*$. Recall our definition of the Kronecker product $\otimes$, where for vectors u, v we have $u \otimes v = uv^\top$. With this notation, we have the following expansion of $\theta_1 - \theta^*$. In these lemmas, $\mathcal{R}_3$ denotes a vector Z for which $\mathbb{E}[\|Z\|_2^2] \le cn^{-3}$ for a numerical constant c.

Lemma 12 *Under the conditions of Theorem 4, we have*

$$\begin{aligned}\theta_1 - \theta^* = &-\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(\nabla^2 F_1(\theta^*) - \Sigma)\Sigma^{-1}\nabla F_1(\theta^*) \\ &- \Sigma^{-1}\nabla^3 F_0(\theta^*)\left((\Sigma^{-1}\nabla F_1(\theta^*)) \otimes (\Sigma^{-1}\nabla F_1(\theta^*))\right) \\ &+ \left(M^2G^6/\lambda^6 + G^4L^2 d\log(d)/\lambda^4\right)\mathcal{R}_3.\end{aligned} \tag{37}$$

We prove Lemma 12 in Appendix G. The lemma requires careful moment control over the expansion $\theta_1 - \theta^*$, leading to some technical difficulty, but is similar in spirit to the results leading to Theorem 1.

An immediately analogous result to Lemma 12 follows for our sub-sampled estimators. Since we use $\lceil rn \rceil$ samples to compute θ_2, the second level estimator, we find

Lemma 13 *Under the conditions of Theorem 4, we have*

$$\begin{aligned}\theta_2 - \theta^* = &-\Sigma^{-1}\nabla F_2(\theta^*) + \Sigma^{-1}(\nabla^2 F_2(\theta^*) - \Sigma)\Sigma^{-1}\nabla F_2(\theta^*) \\ &- \Sigma^{-1}\nabla^3 F_0(\theta^*)\left((\Sigma^{-1}\nabla F_2(\theta^*)) \otimes (\Sigma^{-1}\nabla F_2(\theta^*))\right) \\ &+ r^{-\frac{3}{2}}\left(M^2G^6/\lambda^6 + G^4L^2 d\log(d)/\lambda^4\right)\mathcal{R}_3.\end{aligned}$$

C.2 Bias Correction

Now that we have given Taylor expansions that describe the behaviour of $\theta_1 - \theta^*$ and $\theta_2 - \theta^*$, we can prove Lemmas 10 and 11 (though, as noted earlier, we defer the proof of Lemma 11 to Appendix C.4). The key insight is that expectations of terms involving $\nabla F_2(\theta^*)$ are nearly the same as expectations of terms involving $\nabla F_1(\theta^*)$, except that some corrections for the sampling ratio r are necessary.

We begin by noting that

$$\frac{\theta_1 - r\theta_2}{1-r} - \theta^* = \frac{\theta_1 - \theta^*}{1-r} - r\frac{\theta_2 - \theta^*}{1-r}. \tag{38}$$

In Lemmas 12 and 13, we derived expansions for each of the right hand side terms, and since

$$\mathbb{E}[\Sigma^{-1}\nabla F_1(\theta^*)] = 0 \quad \text{and} \quad \mathbb{E}[\Sigma^{-1}\nabla F_2(\theta^*)] = 0,$$

Lemmas 12 and 13 coupled with the rewritten correction (38) yield

$$\begin{aligned}\mathbb{E}[\theta_1 - \theta^* - r(\theta_2 - \theta^*)] = &-r\mathbb{E}[\Sigma^{-1}(\nabla^2 F_2(\theta^*) - \Sigma)\Sigma^{-1}\nabla F_2(\theta^*)] \\ &+ \mathbb{E}[\Sigma^{-1}(\nabla^2 F_1(\theta^*) - \Sigma)\Sigma^{-1}\nabla F_1(\theta^*)] \\ &+ r\mathbb{E}[\Sigma^{-1}\nabla^3 F_0(\theta^*)\left((\Sigma^{-1}\nabla F_2(\theta^*)) \otimes (\Sigma^{-1}\nabla F_2(\theta^*))\right)] \\ &- \mathbb{E}[\Sigma^{-1}\nabla^3 F_0(\theta^*)\left((\Sigma^{-1}\nabla F_1(\theta^*)) \otimes (\Sigma^{-1}\nabla F_1(\theta^*))\right)] \\ &+ O(1)r^{-1/2}\left(M^2G^6/\lambda^6 + G^4L^2 d\log(d)/\lambda^4\right)n^{-3/2}.\end{aligned} \tag{39}$$

Here the remainder terms follow because of the $r^{-3/2}\mathcal{R}_3$ term on $\theta_2 - \theta^*$.

C.3 Proof of Lemma 10

To prove the claim in the lemma, it suffices to show that

$$r\mathbb{E}[\Sigma^{-1}(\nabla^2 F_2(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_2(\theta^*)] = \mathbb{E}[\Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_1(\theta^*)] \tag{40}$$

and

$$\begin{aligned} &r\mathbb{E}[\Sigma^{-1}\nabla^3 F_0(\theta^*)\left((\Sigma^{-1}\nabla F_2(\theta^*))\otimes(\Sigma^{-1}\nabla F_2(\theta^*))\right)] \\ &\quad = \mathbb{E}[\Sigma^{-1}\nabla^3 F_0(\theta^*)\left((\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}\nabla F_1(\theta^*))\right)] \end{aligned} \tag{41}$$

Indeed, these two claims combined with the expansion (39) yield the bound (35) in Lemma 10 immediately.

We first consider the difference (40). To make things notationally simpler, we define functions $A : \mathcal{X} \to \mathbb{R}^{d\times d}$ and $B : \mathcal{X} \to \mathbb{R}^d$ via $A(x) := \Sigma^{-1}(\nabla^2 f(\theta^*;x)-\Sigma)$ and $B(x) := \Sigma^{-1}\nabla f(\theta^*;x)$. If we let $S_1 = \{X_1,\ldots,X_n\}$ be the original samples and $S_2 = \{Y_1,\ldots,Y_{rn}\}$ be the subsampled data set, we must show

$$r\mathbb{E}\left[\frac{1}{(rn)^2}\sum_{i,j}^{rn} A(Y_i)B(Y_j)\right] = \mathbb{E}\left[\frac{1}{n^2}\sum_{i,j}^{n} A(X_i)B(X_j)\right].$$

Since the Y_i are sampled without replacement (i.e., from P directly), and $\mathbb{E}[A(X_i)] = 0$ and $\mathbb{E}[B(X_i)] = 0$, we find that $\mathbb{E}[A(Y_i)B(Y_j)] = 0$ for $i \neq j$, and thus

$$\sum_{i,j}^{rn}\mathbb{E}[A(Y_i)B(Y_j)] = \sum_{i=1}^{rn}\mathbb{E}[A(Y_i)B(Y_i)] = rn\mathbb{E}[A(Y_1)B(Y_1)].$$

In particular, we see that the equality (40) holds:

$$\begin{aligned} \frac{r}{(rn)^2}\sum_{i,j}^{rn}\mathbb{E}[A(Y_i)B(Y_j)] = \frac{r}{rn}\mathbb{E}[A(Y_1)B(Y_1)] &= \frac{1}{n}\mathbb{E}[A(X_1)B(X_1)] \\ &= \frac{1}{n^2}\sum_{i,j}^{n}\mathbb{E}[A(X_i)B(X_j)]. \end{aligned}$$

The statement (41) follows from analogous arguments.

C.4 Proof of Lemma 11

The proof of Lemma 11 follows from that of Lemmas 12 and 13. We first claim that

$$\theta_1 - \theta^* = -\Sigma^{-1}\nabla F_1(\theta^*) + \mathcal{R}_2 \quad \text{and} \quad \theta_2 - \theta^* = -\Sigma^{-1}\nabla F_2(\theta^*) + r^{-1}\mathcal{R}_2. \tag{42}$$

The proofs of both claims similar, so we focus on proving the second statement. Using the inequality $(a+b+c)^2 \leq 3(a^2+b^2+c^2)$ and Lemma 13, we see that

$$\begin{aligned} \mathbb{E}\left[\left\|\theta_2 - \theta^* + \Sigma^{-1}\nabla F_2(\theta^*)\right\|_2^2\right] &\leq 3\mathbb{E}\left[\left\|\Sigma^{-1}(\nabla^2 F_2(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_2(\theta^*)\right\|_2^2\right] \\ &\quad + 3\mathbb{E}\left[\left\|\Sigma^{-1}\nabla^3 F_0(\theta^*)\left((\Sigma^{-1}\nabla F_2(\theta^*))\otimes(\Sigma^{-1}\nabla F_2(\theta^*))\right)\right\|_2^2\right] \\ &\quad + 3r^{-3}O(n^{-3}). \end{aligned} \tag{43}$$

We now bound the first two terms in inequality (43). Applying the Cauchy-Schwarz inequality and Lemma 7, the first term can be upper bounded as

$$\begin{aligned}
&\mathbb{E}\left[\left\|\Sigma^{-1}(\nabla^2 F_2(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_2(\theta^*)\right\|_2^2\right] \\
&\leq \left(\mathbb{E}\left[\left|\left|\left|\Sigma^{-1}(\nabla^2 F_2(\theta^*)-\Sigma)\right|\right|\right|_2^4\right]\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_2(\theta^*)\right\|_2^4\right]\right)^{1/2} \\
&= \left(r^{-2}\right)O(\log^2(d)n^{-2})\cdot r^{-2}O(n^{-2})\big)^{1/2} \;=\; r^{-2}O(n^{-2}),
\end{aligned}$$

where the order notation subsumes the logarithmic factor in the dimension. Since $\nabla^3 F_0(\theta^*):\mathbb{R}^{d^2}\to\mathbb{R}^d$ is linear, the second term in the inequality (43) may be bounded completely analogously as it involves the outer product $\Sigma^{-1}\nabla F_2(\theta^*)\otimes\Sigma^{-1}\nabla F_2(\theta^*)$. Recalling the bound (43), we have thus shown that

$$\mathbb{E}\left[\left\|\theta_2-\theta^*+\Sigma^{-1}\nabla F_2(\theta^*)\right\|_2^2\right]=r^{-2}O(n^{-2}),$$

or $\theta_2-\theta^*=-\Sigma^{-1}\nabla F_2(\theta^*)+r^{-1}\mathcal{R}_2$. The proof of the first equality in Equation (42) is entirely analogous.

We now apply the equalities (42) to obtain the result of the lemma. We have

$$\mathbb{E}\left[\|\theta_1-\theta^*-r(\theta_2-\theta^*)\|_2^2\right]=\mathbb{E}\left[\left\|-\Sigma^{-1}\nabla F_1(\theta^*)+r\Sigma^{-1}\nabla F_2(\theta^*)+\mathcal{R}_2\right\|_2^2\right].$$

Using the inequality $(a+b)^2\leq(1+\eta)a^2+(1+1/\eta)b^2$ for any $\eta\geq 0$, we have

$$\begin{aligned}
(a+b+c)^2 &\leq (1+\eta)a^2+(1+1/\eta)(b+c)^2 \\
&\leq (1+\eta)a^2+(1+1/\eta)(1+\alpha)b^2+(1+1/\eta)(1+1/\alpha)c^2
\end{aligned}$$

for any $\eta,\alpha\geq 0$. Taking $\eta=1$ and $\alpha=1/2$, we obtain $(a+b+c)^2\leq 2a^2+3b^2+6c^2$, so applying the triangle inequality, we have

$$\begin{aligned}
\mathbb{E}\left[\|\theta_1-\theta^*-r(\theta_2-\theta^*)\|_2^2\right] &= \mathbb{E}\left[\left\|-\Sigma^{-1}\nabla F_1(\theta^*)+r\Sigma^{-1}\nabla F_2(\theta^*)+\mathcal{R}_2\right\|_2^2\right] \\
&\leq 2\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right]+3r^2\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_2(\theta^*)\right\|_2^2\right]+O(n^{-2}).
\end{aligned} \tag{44}$$

Since F_2 is a sub-sampled version of F_1, algebraic manipulations yield

$$\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_2(\theta^*)\right\|_2^2\right]=\frac{n}{rn}\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right]=\frac{1}{r}\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^2\right]. \tag{45}$$

Combining equations (44) and (45), we obtain the desired bound (36).

Appendix D. Proof of Theorem 5

We begin by recalling that if θ^n denotes the output of performing stochastic gradient on one machine, then from the inequality (19) we have the upper bound

$$\mathbb{E}[\left\|\overline{\theta}^n-\theta^*\right\|_2^2]\leq\frac{1}{m}\mathbb{E}[\|\theta^n-\theta^*\|_2^2]+\|\mathbb{E}[\theta^n-\theta^*]\|_2^2.$$

To prove the error bound (15), it thus suffices to prove the inequalities

$$\mathbb{E}[\|\theta^n - \theta^*\|_2^2] \le \frac{\alpha G^2}{\lambda^2 n}, \quad \text{and} \tag{46}$$

$$\|\mathbb{E}[\theta^n - \theta^*]\|_2^2 \le \frac{\beta^2}{n^{3/2}}. \tag{47}$$

Before proving the theorem, we introduce some notation and a few preliminary results. Let $g_t = \nabla f(\theta^t; X_t)$ be the gradient of the t^{th} sample in stochastic gradient descent, where we consider running SGD on a single machine. We also let

$$\Pi(v) := \operatorname*{argmin}_{\theta \in \Theta} \left\{ \|\theta - v\|_2^2 \right\}$$

denote the projection of the point v onto the domain Θ.

We now state a known result, which gives sharp rates on the convergence of the iterates $\{\theta^t\}$ in stochastic gradient descent.

Lemma 14 (Rakhlin et al., 2012) *Assume that $\mathbb{E}[\|g_t\|_2^2] \le G^2$ for all t. Choosing $\eta_t = \frac{c}{\lambda t}$ for some $c \ge 1$, for any $t \in \mathbb{N}$ we have*

$$\mathbb{E}\left[\left\|\theta^t - \theta^*\right\|_2^2\right] \le \frac{\alpha G^2}{\lambda^2 t} \quad \textit{where} \quad \alpha = 4c^2.$$

With these ingredients, we can now turn to the proof of Theorem 5. Lemma 14 gives the inequality (46), so it remains to prove that $\overline{\theta}^n$ has the smaller bound (47) on its bias. To that end, recall the neighborhood $U_\rho \subset \Theta$ in Assumption 5, and note that

$$\begin{aligned}
\theta^{t+1} - \theta^* &= \Pi(\theta^t - \eta_t g_t - \theta^*) \\
&= \theta^t - \eta_t g_t - \theta^* + 1_{\left(\theta^{t+1} \notin U_\rho\right)} \left(\Pi(\theta^t - \eta_t g_t) - (\theta^t - \eta_t g_t)\right)
\end{aligned}$$

since when $\theta \in U_\rho$, we have $\Pi(\theta) = \theta$. Consequently, an application of the triangle inequality gives

$$\left\|\mathbb{E}[\theta^{t+1} - \theta^*]\right\|_2 \le \left\|\mathbb{E}[\theta^t - \eta_t g_t - \theta^*]\right\|_2 + \mathbb{E}[\left\|(\Pi(\theta^t - \eta_t g_t) - (\theta^t - \eta_t g_t))1(\theta^{t+1} \notin U_\rho)\right\|_2].$$

By the definition of the projection and the fact that $\theta^t \in \Theta$, we additionally have

$$\left\|\Pi(\theta^t - \eta_t g_t) - (\theta^t - \eta_t g_t)\right\|_2 \le \left\|\theta^t - (\theta^t - \eta_t g_t))\right\|_2 \le \eta_t \left\|g_t\right\|_2.$$

Thus, by applying Hölder's inequality (with the conjugate choices $(p, q) = (4, \frac{4}{3})$) and Assumption 5, we have

$$\begin{aligned}
\left\|\mathbb{E}[\theta^{t+1} - \theta^*]\right\|_2 &\le \left\|\mathbb{E}[\theta^t - \eta_t g_t - \theta^*]\right\|_2 + \eta_t \mathbb{E}[\|g_t\|_2 1_{\left(\theta^{t+1} \notin U_\rho\right)}] \\
&\le \left\|\mathbb{E}[\theta^t - \eta_t g_t - \theta^*]\right\|_2 + \eta_t \sqrt[4]{\mathbb{E}[\|g_t\|_2^4]} \left(\mathbb{E}[1^{4/3}_{\left(\theta^t \notin U_\rho\right)}]\right)^{3/4} \\
&\le \left\|\mathbb{E}[\theta^t - \eta_t g_t - \theta^*]\right\|_2 + \eta_t G \left(\mathbb{P}(\theta^t \notin U_\rho)\right)^{3/4} \\
&\le \left\|\mathbb{E}[\theta^t - \eta_t g_t - \theta^*]\right\|_2 + \eta_t G \left(\frac{\mathbb{E}\left\|\theta^{t+1} - \theta^*\right\|_2^2}{\rho^2}\right)^{3/4},
\end{aligned} \tag{48}$$

the inequality (48) following from an application of Markov's inequality. By applying Lemma 14, we finally obtain

$$\begin{aligned}\left\|\mathbb{E}[\theta^{t+1}-\theta^*]\right\|_2 &\leq \left\|\mathbb{E}[\theta^t-\eta_t g_t-\theta^*]\right\|_2+\eta_t G\left(\frac{\alpha G^2}{\lambda^2\rho^2 t}\right)^{3/4}\\ &= \left\|\mathbb{E}[\theta^t-\eta_t g_t-\theta^*]\right\|_2+\frac{c\alpha^{3/4}G^{5/2}}{\lambda^{5/2}\rho^{3/2}}\cdot\frac{1}{t^{7/4}}. \end{aligned} \tag{49}$$

Now we turn to controlling the rate at which $\theta^t-\eta_t g_t$ goes to zero. Let $f_t(\cdot)=f(\cdot;X_t)$ be shorthand for the loss evaluated on the t^{th} data point. By defining

$$r_t = g_t-\nabla f_t(\theta^*)-\nabla^2 f_t(\theta^*)(\theta^t-\theta^*),$$

a bit of algebra yields

$$g_t=\nabla f_t(\theta^*)+\nabla^2 f_t(\theta^*)(\theta^t-\theta^*)+r_t.$$

Since θ^t belongs to the σ-field of $X_1,\ldots,X_{t-1}$, the Hessian $\nabla^2 f_t(\theta^*)$ is (conditionally) independent of θ^t and

$$\mathbb{E}[g_t]=\nabla^2 F_0(\theta^*)\mathbb{E}[\theta^t-\theta^*]+\mathbb{E}[r_t 1_{\left(\theta^t\in U_\rho\right)}]+\mathbb{E}[r_t 1_{\left(\theta^t\notin U_\rho\right)}]. \tag{50}$$

If $\theta^t\in U_\rho$, then Taylor's theorem implies that r_t is the Lagrange remainder

$$r_t=(\nabla^2 f_t(\theta')-\nabla^2 f_t(\theta^*))(\theta'-\theta^*),$$

where $\theta'=\kappa\theta^t+(1-\kappa)\theta^*$ for some $\kappa\in[0,1]$. Applying Assumption 5 and Hölder's inequality, we find that since θ^t is conditionally independent of X_t,

$$\begin{aligned}\mathbb{E}\left[\left\|r_t 1_{\left(\theta^t\in U_\rho\right)}\right\|_2\right] &\leq \mathbb{E}\left[\left|\left|\left|\nabla^2 f(\theta';X_t)-\nabla^2 f(\theta^*;X_t)\right|\right|\right|\left\|\theta^t-\theta^*\right\|_2 1_{\left(\theta^t\in U_\rho\right)}\right]\\ &\leq \mathbb{E}\left[L(X_t)\left\|\theta^t-\theta^*\right\|_2^2\right]=\mathbb{E}[L(X_t)]\mathbb{E}[\left\|\theta^t-\theta^*\right\|_2^2]\\ &\leq L\mathbb{E}\left[\left\|\theta^t-\theta^*\right\|_2^2\right]\leq\frac{\alpha L G^2}{\lambda^2 t}.\end{aligned}$$

On the other hand, when $\theta^t\notin U_\rho$, we have the following sequence of inequalities:

$$\begin{aligned}\mathbb{E}\left[\left\|r_t 1_{\left(\theta^t\notin U_\rho\right)}\right\|_2\right] &\overset{(i)}{\leq} \sqrt[4]{\mathbb{E}[\|r_t\|_2^4]}\left(\mathbb{P}(\theta^t\notin U_\rho)\right)^{3/4}\\ &\overset{(ii)}{\leq} \sqrt[4]{3^3\left(\mathbb{E}[\|g_t\|_2^4]+\mathbb{E}[\|\nabla f_t(\theta^*)\|_2^4]+\mathbb{E}[\|\nabla^2 f_t(\theta^*)(\theta^t-\theta^*)\|_2^4]\right)}\left(\mathbb{P}(\theta^t\notin U_\rho)\right)^{3/4}\\ &\leq 3^{3/4}\sqrt[4]{G^4+G^4+H^4R^4}\left(\mathbb{P}(\theta^t\notin U_\rho)\right)^{3/4}\\ &\overset{(iii)}{\leq} 3(G+HR)\left(\frac{\alpha G^2}{\lambda^2\rho^2 t}\right)^{3/4}.\end{aligned}$$

Here step (i) follows from Hölder's inequality (again applied with the conjugates $(p,q)=(4,\frac{4}{3})$); step (ii) follows from Jensen's inequality, since $(a+b+c)^4\leq 3^3(a^4+b^4+c^4)$; and step (iii) follows

from Markov's inequality, as in the bounds (48) and (49). Combining our two bounds on r_t, we find that

$$\mathbb{E}[\|r_t\|_2] \leq \frac{\alpha L G^2}{\lambda^2 t} + \frac{3\alpha^{3/4} G^{3/2}(G+HR)}{\lambda^{3/2}\rho^{3/2}} \cdot \frac{1}{t^{3/4}}. \tag{51}$$

By combining the expansion (50) with the bound (51), we find that

$$\begin{aligned}\left\|\mathbb{E}[\theta^t - \eta_t g_t - \theta^*]\right\|_2 &= \left\|\mathbb{E}[(I - \eta_t \nabla^2 F_0(\theta^*))(\theta^t - \theta^*) + \eta_t r_t]\right\|_2 \\ &\leq \left\|\mathbb{E}[(I - \eta_t \nabla^2 F_0(\theta^*))(\theta^t - \theta^*)]\right\|_2 + \frac{c\alpha L G^2}{\lambda^3 t^2} + \frac{3c\alpha^{3/4} G^{3/2}(G+HR)}{\lambda^{5/2}\rho^{3/2}} \cdot \frac{1}{t^{7/4}}.\end{aligned}$$

Using the earlier bound (49), this inequality then yields

$$\left\|\mathbb{E}[\theta^{t+1} - \theta^*]\right\|_2 \leq \left|\left|\left|I - \eta_t \nabla^t F_0(\theta^*)\right|\right|\right| \left\|\mathbb{E}[\theta^t - \theta^*]\right\|_2 + \frac{c\alpha^{3/4} G^{3/2}}{\lambda^{5/2} t^{7/4}} \left(\frac{\alpha^{1/4} L G^{1/2}}{\lambda^{1/2} t^{1/4}} + \frac{4G + HR}{\rho^{3/2}}\right).$$

We now complete the proof via an inductive argument using our immediately preceding bounds. Our reasoning follows a similar induction given by Rakhlin et al. (2012). First, note that by strong convexity and our condition that $\left|\left|\left|\nabla^2 F_0(\theta^*)\right|\right|\right| \leq H$, we have

$$\left|\left|\left|I - \eta_t \nabla^2 F_0(\theta^*)\right|\right|\right| = 1 - \eta_t \lambda_{\min}(\nabla^2 F_0(\theta^*) \leq 1 - \eta_t \lambda$$

whenever $1 - \eta_t H \geq 0$. Define $\tau_0 = \lceil cH/\lambda \rceil$; then for $t \geq t_0$ we obtain

$$\left\|\mathbb{E}[\theta^{t+1} - \theta^*]\right\|_2 \leq (1 - c/t) \left\|\mathbb{E}[\theta^t - \theta^*]\right\|_2 + \frac{1}{t^{7/4}} \cdot \frac{c\alpha^{3/4} G^{3/2}}{\lambda^{5/2}} \left(\frac{\alpha^{1/4} L G^{1/2}}{\lambda^{1/2} t^{1/4}} + \frac{4G + HR}{\rho^{3/2}}\right). \tag{52}$$

For shorthand, we define two intermediate variables

$$a_t = \left\|\mathbb{E}(\theta^t - \theta^*)\right\|_2 \quad \text{and} \quad b = \frac{c\alpha^{3/4} G^{3/2}}{\lambda^{5/2}} \left(\frac{\alpha^{1/4} L G^{1/2}}{\lambda^{1/2}} + \frac{4G + HR}{\rho^{3/2}}\right).$$

Inequality (52) then implies the inductive relation $a_{t+1} \leq (1 - c/t) a_t + b/t^{7/4}$. Now we show that by defining $\beta = \max\{\tau_0 R, b/(c-1)\}$, we have $a_t \leq \beta/t^{3/4}$. Indeed, it is clear that $a_1 \leq \tau_0 R$. Using the inductive hypothesis, we then have

$$a_{t+1} \leq \frac{(1 - c/t)\beta}{t^{3/4}} + \frac{b}{t^{7/4}} = \frac{\beta(t-1)}{t^{7/4}} - \frac{\beta(c-1) - b}{t^2} \leq \frac{\beta(t-1)}{t^{7/4}} \leq \frac{\beta}{(t+1)^{3/4}}.$$

This completes the proof of the inequality (47). ■

D.0.1 REMARK

If we assume kth moment bounds instead of 4th, that is, $\mathbb{E}[\left|\left|\left|\nabla^2 f(\theta^*; X)\right|\right|\right|_2^k] \leq H^k$ and $\mathbb{E}[\|g_t\|_2^k] \leq G^k$, we find the following analogue of the bound (52):

$$\begin{aligned}\left\|\mathbb{E}[\theta^{t+1} - \theta^*]\right\|_2 \leq (1 - c/t) &\left\|\mathbb{E}[\theta^t - \theta^*]\right\|_2 \\ &+ \frac{1}{t^{\frac{2k-1}{k}}} \cdot \frac{c\alpha^{\frac{k-1}{k}} G^{\frac{2k-2}{k}}}{\lambda^{\frac{3k-2}{k}}} \left[\frac{\left(54^{1/k} + 1\right) G + 54^{1/k} HR}{\rho^{\frac{2k-2}{k}}} + \frac{\alpha^{1/k} L G^{2/k}}{\lambda^{2/k} t^{1/k}}\right].\end{aligned}$$

In this case, if we define

$$b = \frac{c\alpha^{\frac{k-1}{k}} G^{\frac{2k-2}{k}}}{\lambda^{\frac{3k-2}{k}}} \left[\frac{\left(54^{1/k}+1\right) G + 54^{1/k} HR}{\rho^{\frac{2k-2}{k}}} + \frac{\alpha^{1/k} L G^{2/k}}{\lambda^{2/k}} \right] \quad \text{and} \quad \beta = \max\left\{ \tau_0 R, \frac{b}{c-1} \right\},$$

we have the same result except we obtain the bound $\|\mathbb{E}[\theta^n - \theta^*]\|_2^2 \leq \beta^2 / n^{\frac{2k-2}{k}}$.

Appendix E. Proof of Lemma 6

We first prove that under the conditions given in the lemma statement, the function F_1 is $(1-\rho)\lambda$-strongly convex over the ball $U := \left\{\theta \in \mathbb{R}^d : \|\theta - \theta^*\|_2 < \delta_\rho\right\}$ around θ^*. Indeed, fix $\gamma \in U$, then use the triangle inequality to conclude that

$$\begin{aligned} \left|\left|\left|\nabla^2 F_1(\gamma) - \nabla^2 F_0(\theta^*)\right|\right|\right|_2 &\leq \left|\left|\left|\nabla^2 F_1(\gamma) - \nabla^2 F_1(\theta^*)\right|\right|\right|_2 + \left|\left|\left|\nabla^2 F_1(\theta^*) - \nabla^2 F_0(\theta^*)\right|\right|\right|_2 \\ &\leq L \|\gamma - \theta^*\|_2 + \frac{\rho\lambda}{2}. \end{aligned}$$

Here we used Assumption 3 on the first term and the fact that the event $\mathcal{E}_1$ holds on the second. By our choice of $\delta_\rho \leq \rho\lambda/4L$, this final term is bounded by $\lambda\rho$. In particular, we have

$$\nabla^2 F_0(\theta^*) \succeq \lambda I \quad \text{so} \quad \nabla^2 F_1(\gamma) \succeq \lambda I - \rho\lambda I = (1-\rho)\lambda I,$$

which proves that F_1 is $(1-\rho)\lambda$-strongly convex on the ball U.

In order to prove the conclusion of the lemma, we argue that since F_1 is (locally) strongly convex, if the function F_1 has small gradient at the point θ^*, it must be the case that the minimizer θ_1 of F_1 is near θ^*. Then we can employ reasoning similar to standard analyses of optimality for globally strongly convex functions (e.g., Boyd and Vandenberghe, 2004, Chapter 9). By definition of (the local) strong convexity on the set U, for any $\theta' \in \Theta$, we have

$$F_1(\theta') \geq F_1(\theta^*) + \left\langle \nabla F_1(\theta^*), \theta' - \theta^* \right\rangle + \frac{(1-\rho)\lambda}{2} \min\left\{ \left\|\theta^* - \theta'\right\|_2^2, \delta_\rho^2 \right\}.$$

Rewriting this inequality, we find that

$$\begin{aligned} \min\left\{ \left\|\theta^* - \theta'\right\|_2^2, \delta_\rho^2 \right\} &\leq \frac{2}{(1-\rho)\lambda} \left[F_1(\theta') - F_1(\theta^*) + \left\langle \nabla F_1(\theta^*), \theta' - \theta^* \right\rangle \right] \\ &\leq \frac{2}{(1-\rho)\lambda} \left[F_1(\theta') - F_1(\theta^*) + \|\nabla F_1(\theta^*)\|_2 \left\|\theta' - \theta^*\right\|_2 \right]. \end{aligned}$$

Dividing each side by $\|\theta' - \theta^*\|_2$, then noting that we may set $\theta' = \kappa\theta_1 + (1-\kappa)\theta^*$ for any $\kappa \in [0,1]$, we have

$$\min\left\{ \kappa \|\theta_1 - \theta^*\|_2, \frac{\delta_\rho^2}{\kappa \|\theta_1 - \theta^*\|_2} \right\} \leq \frac{2\left[F_1(\kappa\theta_1 + (1-\kappa)\theta^*) - F_1(\theta^*)\right]}{\kappa(1-\rho)\lambda \|\theta_1 - \theta^*\|_2} + \frac{2\|\nabla F_1(\theta^*)\|_2}{(1-\rho)\lambda}.$$

Of course, $F_1(\theta_1) < F_1(\theta^*)$ by assumption, so we find that for any $\kappa \in (0,1)$ we have the strict inequality

$$\min\left\{ \kappa \|\theta_1 - \theta^*\|_2, \frac{\delta_\rho^2}{\kappa \|\theta_1 - \theta^*\|_2} \right\} < \frac{2\|\nabla F_1(\theta^*)\|_2}{(1-\rho)\lambda} \leq \delta_\rho,$$

the last inequality following from the definition of $\mathcal{E}_2$. Since this holds for any $\kappa \in (0,1)$, if $\|\theta_1 - \theta^*\|_2 > \delta_\rho$, we may set $\kappa = \delta_\rho / \|\theta_1 - \theta^*\|_2$, which would yield a contradiction. Thus, we have $\|\theta_1 - \theta^*\|_2 \leq \delta_\rho$, and by our earlier inequalities,

$$\|\theta_1 - \theta^*\|_2^2 \leq \frac{2}{(1-\rho)\lambda}\left[F_1(\theta_1) - F_1(\theta^*) + \|\nabla F_1(\theta^*)\|_2 \|\theta_1 - \theta^*\|_2\right] \leq \frac{2\|\nabla F_1(\theta^*)\|_2}{(1-\rho)\lambda}\|\theta_1 - \theta^*\|_2.$$

Dividing by $\|\theta_1 - \theta^*\|_2$ completes the proof. ■

Appendix F. Moment Bounds

In this appendix, we state two useful moment bounds, showing how they combine to provide a proof of Lemma 7. The two lemmas are a vector and a non-commutative matrix variant of the classical Rosenthal inequalities. We begin with the case of independent random vectors:

Lemma 15 (de Acosta, 1981, Theorem 2.1) *Let $k \geq 2$ and X_i be a sequence of independent random vectors in a separable Banach space with norm $\|\cdot\|$ and $\mathbb{E}[\|X_i\|^k] < \infty$. There exists a finite constant C_k such that*

$$\mathbb{E}\left[\left|\left\|\sum_{i=1}^n X_i\right\| - \mathbb{E}\left[\left\|\sum_{i=1}^n X_i\right\|\right]\right|^k\right] \leq C_k\left[\left(\sum_{i=1}^n \mathbb{E}[\|X_i\|^2]\right)^{k/2} + \sum_{i=1}^n \mathbb{E}[\|X_i\|^k]\right].$$

We say that a random matrix X is symmetrically distributed if X and $-X$ have the same distribution. For such matrices, we have:

Lemma 16 (Chen et al., 2012, Theorem A.1(2)) *Let $X_i \in \mathbb{R}^{d\times d}$ be independent and symmetrically distributed Hermitian matrices. Then*

$$\mathbb{E}\left[\left\|\sum_{i=1}^n X_i\right\|^k\right]^{1/k} \leq \sqrt{2e\log d}\left\|\left(\sum_{i=1}^n \mathbb{E}\left[X_i^2\right]\right)^{1/2}\right\| + 2e\log d\left(\mathbb{E}[\max_i \|X_i\|^k]\right)^{1/k}.$$

Equipped with these two auxiliary results, we turn to our proof Lemma 7. To prove the first bound (24), let $2 \leq k \leq k_0$ and note that by Jensen's inequality, we have

$$\mathbb{E}[\|\nabla F_1(\theta^*)\|_2^k] \leq 2^{k-1}\mathbb{E}\left[\left|\,\|\nabla F_1(\theta^*)\|_2 - \mathbb{E}[\|\nabla F_1(\theta^*)\|_2]\right|^k\right] + 2^{k-1}\mathbb{E}\left[\|\nabla F_1(\theta^*)\|_2\right]^k.$$

Again applying Jensen's inequality, $\mathbb{E}[\|\nabla f(\theta^*;X)\|_2^2] \leq G^2$. Thus by recalling the definition $\nabla F_1(\theta^*) = \frac{1}{n}\sum_{i=1}^n \nabla f(\theta^*;X_i)$ and applying the inequality

$$\mathbb{E}[\|\nabla F_1(\theta^*)\|_2] \leq \mathbb{E}[\|\nabla F_1(\theta^*)\|_2^2]^{1/2} \leq n^{-1/2}G,$$

we see that Lemma 15 implies $\mathbb{E}\left[\|\nabla F_1(\theta^*)\|_2^k\right]$ is upper bounded by

$$2^{k-1}C_k\left[\left(\frac{1}{n^2}\sum_{i=1}^n \mathbb{E}[\|\nabla f(\theta;X_i)\|_2^2]\right)^{k/2} + \frac{1}{n^k}\sum_{i=1}^n \mathbb{E}[\|\nabla f(\theta^*;X_i)\|_2^k]\right] + 2^{k-1}\mathbb{E}[\|\nabla F_1(\theta^*)\|_2]^k$$
$$\leq 2^{k-1}\frac{C_k}{n^{k/2}}\left[\left(\frac{1}{n}\sum_{i=1}^n \mathbb{E}[\|\nabla f(\theta^*;X_i)\|_2^2]\right)^{k/2} + \frac{1}{n^{k/2}}\sum_{i=1}^n \mathbb{E}[\|\nabla f(\theta^*;X_i)\|_2^k]\right] + \frac{2^{k-1}G^k}{n^{k/2}}.$$

Applying Jensen's inequality yields

$$\left(\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\|\nabla f(\theta^*;X_i)\|_2^2]\right)^{k/2} \leq \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\|\nabla f(\theta^*;X_i)\|_2^2]^{k/2} \leq G^k,$$

completes the proof of the inequality (24).

The proof of the bound (25) requires a very slightly more delicate argument involving symmetrization step. Define matrices $Z_i = \frac{1}{n}\left(\nabla^2 f(\theta^*;X_i) - \nabla^2 F_0(\theta^*)\right)$. If $\varepsilon_i \in \{\pm 1\}$ are i.i.d. Rademacher variables independent of Z_i, then for any integer k in the interval $[2,k_2]$, a standard symmetrization argument (e.g., Ledoux and Talagrand, 1991, Lemma 6.3) implies that

$$\mathbb{E}\left[\left|\left|\left|\sum_{i=1}^{n} Z_i\right|\right|\right|^k\right]^{1/k} \leq 2\mathbb{E}\left[\left|\left|\left|\sum_{i=1}^{n}\varepsilon_i Z_i\right|\right|\right|^k\right]^{1/k}.$$

Now we may apply Lemma 16, since the matrices $\varepsilon_i Z_i$ are Hermitian and symmetrically distributed; by expanding the definition of the Z_i, we find that

$$\mathbb{E}\left[\left|\left|\left|\nabla^2 F_1(\theta^*) - \nabla^2 F_0(\theta^*)\right|\right|\right|^k\right]^{1/k} \leq 5\sqrt{\log d}\left|\left|\left|\left(\frac{1}{n^2}\sum_{i=1}^{n}\mathbb{E}[(\nabla^2 f(\theta;X_i) - \nabla^2 F_0(\theta^*))^2]\right)^{1/2}\right|\right|\right| + 4e\log d\left(n^{-k}\mathbb{E}[\max_i \left|\left|\left|\nabla^2 f(\theta^*;X_i) - \nabla^2 F_0(\theta^*)\right|\right|\right|^k]\right)^{1/k}.$$

Since the X_i are i.i.d., we have

$$\left|\left|\left|\left(\frac{1}{n^2}\sum_{i=1}^{n}\mathbb{E}[(\nabla^2 f(\theta;X_i) - \nabla^2 F_0(\theta^*))^2]\right)^{1/2}\right|\right|\right| = \left|\left|\left|n^{-1/2}\mathbb{E}\left[\left(\nabla^2 f(\theta^*;X) - \nabla^2 F_0(\theta^*)\right)^2\right]^{1/2}\right|\right|\right| \leq n^{-1/2}\mathbb{E}\left[\left|\left|\left|\nabla^2 f(\theta^*;X) - \nabla^2 F_0(\theta^*)\right|\right|\right|^2\right]^{1/2}$$

by Jensen's inequality, since $\left|\left|\left|A^{1/2}\right|\right|\right| = |||A|||^{1/2}$ for semidefinite A. Finally, noting that

$$\frac{1}{n^k}\mathbb{E}\left[\max_i \left|\left|\left|\nabla^2 f(\theta^*;X_i) - \nabla^2 F_0(\theta^*)\right|\right|\right|^k\right] \leq \frac{n}{n^k}\mathbb{E}\left[\left|\left|\left|\nabla^2 f(\theta^*;X) - \nabla^2 F_0(\theta^*)\right|\right|\right|^k\right] \leq n^{1-k}H^k$$

completes the proof of the second bound (25).

Appendix G. Proof of Lemma 12

The proof follows from a slightly more careful application of the Taylor expansion (21). The starting point in our proof is to recall the success events (20) and the joint event $\mathcal{E} := \mathcal{E}_0 \cap \mathcal{E}_1 \cap \mathcal{E}_2$. We begin by arguing that we may focus on the case where $\mathcal{E}$ holds. Let C denote the right hand side of the equality (37) except for the remainder $\mathcal{R}_3$ term. By Assumption 3, we follow the bound (26) (with $\min\{k_0,k_1,k_2\} \geq 8$) to find that

$$\mathbb{E}\left[1_{(\mathcal{E}^c)}\|\theta_1 - \theta^*\|_2^2\right] = O\left(R^2 n^{-4}\right),$$

so we can focus on the case where the joint event $\mathcal{E} = \mathcal{E}_0 \cap \mathcal{E}_1 \cap \mathcal{E}_2$ does occur.

Defining $\Delta = \theta_1 - \theta^*$ for notational convenience, on $\mathcal{E}$ we have that for some $\kappa \in [0,1]$, with $\theta' = (1-\kappa)\theta_1 + \kappa\theta^*$,

$$\begin{aligned} 0 &= \nabla F_1(\theta^*) + \nabla^2 F_1(\theta^*)\Delta + \nabla^3 F_1(\theta')(\Delta \otimes \Delta) \\ &= \nabla F_1(\theta^*) + \nabla^2 F_0(\theta^*)\Delta + \nabla^3 F_0(\theta^*)(\Delta \otimes \Delta) \\ &\quad + (\nabla^2 F_1(\theta^*) - \nabla^2 F_0(\theta^*))\Delta + (\nabla^3 F_1(\theta') - \nabla^3 F_0(\theta^*))(\Delta \otimes \Delta). \end{aligned}$$

Now, we recall the definition $\Sigma = \nabla^2 F_0(\theta^*)$, the Hessian of the risk at the optimal point, and solve for the error Δ to see that

$$\begin{aligned} \Delta = &-\Sigma^{-1}\nabla F_1(\theta^*) - \Sigma^{-1}(\nabla^2 F_1(\theta^*) - \Sigma)\Delta - \Sigma^{-1}\nabla^3 F_1(\theta^*)(\Delta \otimes \Delta) \\ &+ \Sigma^{-1}(\nabla^3 F_0(\theta^*) - \nabla^3 F_1(\theta'))(\Delta \otimes \Delta) \end{aligned} \tag{53}$$

on the event $\mathcal{E}$. As we did in the proof of Theorem 1, specifically in deriving the recursive equality (28), we may apply the expansion (23) of $\Delta = \theta_1 - \theta^*$ to obtain a clean asymptotic expansion of Δ using (53). Recall the definition $P = \nabla^2 F_0(\theta^*) - \nabla^2 F_1(\theta^*)$ for shorthand here (as in the expansion (23), though we no longer require Q).

First, we claim that

$$1_{(\mathcal{E})}(\nabla^3 F_0(\theta^*) - \nabla^3 F_1(\theta'))(\Delta \otimes \Delta) = \left(M^2 G^6/\lambda^6 + G^4 L^2 d \log(d)/\lambda^4\right) \mathcal{R}_3. \tag{54}$$

To prove the above expression, we add and subtract $\nabla^3 F_1(\theta^*)$ (and drop $1_{(\mathcal{E})}$ for simplicity). We must control

$$(\nabla^3 F_0(\theta^*) - \nabla^3 F_1(\theta^*))(\Delta \otimes \Delta) + (\nabla^3 F_1(\theta^*) - \nabla^3 F_1(\theta'))(\Delta \otimes \Delta).$$

To begin, recall that $|||u \otimes v|||_2 = \left|\left|\left|uv^\top\right|\right|\right|_2 = \|u\|_2 \|v\|_2$. By Assumption 4, on the event $\mathcal{E}$ we have that $\nabla^3 F_1$ is $(1/n)\sum_{i=1}^n M(X_i)$-Lipschitz, so defining $M_n = (1/n)\sum_{i=1}^n M(X_i)$, we have

$$\begin{aligned} \mathbb{E}\left[1_{(\mathcal{E})} \left\|\left(\nabla^3 F_1(\theta^*) - \nabla^3 F_1(\theta')\right)(\Delta \otimes \Delta)\right\|_2^2\right] &\le \mathbb{E}\left[M_n^2 \left\|\theta^* - \theta'\right\|_2^2 \|\Delta\|_2^4\right] \\ &\le \mathbb{E}\left[M_n^8\right]^{1/4} \mathbb{E}\left[\|\theta_1 - \theta^*\|_2^8\right]^{3/4} \le O(1) M^2 \frac{G^6}{\lambda^6 n^3} \end{aligned}$$

by Hölder's inequality and Lemma 8. The remaining term we must control is the derivative difference $\mathbb{E}[\|(\nabla^3 F_1(\theta^*) - \nabla^3 F_0(\theta^*))(\Delta \otimes \Delta)\|_2^2]$. Define the random vector-valued function $\mathsf{G} = \nabla(F_1 - F_0)$, and let G_j denote its jth coordinate. Then by definition we have

$$(\nabla^3 F_1(\theta^*) - \nabla^3 F_0(\theta^*))(\Delta \otimes \Delta) = \left[\Delta^\top(\nabla^2 \mathsf{G}_1(\theta^*))\Delta \;\; \cdots \;\; \Delta^\top(\nabla^2 \mathsf{G}_d(\theta^*))\Delta\right]^\top \in \mathbb{R}^d.$$

Therefore, by the Cauchy-Schwarz inequality and the fact that $x^\top A x \le |||A|||_2 \|x\|_2^2$,

$$\begin{aligned} \mathbb{E}\left[\left\|(\nabla^3 F_1(\theta^*) - \nabla^3 F_0(\theta^*))(\Delta \otimes \Delta)\right\|_2^2\right] &= \sum_{j=1}^d \mathbb{E}\left[\left(\Delta^\top(\nabla^2 \mathsf{G}_j(\theta^*))\Delta\right)^2\right] \\ &\le \sum_{j=1}^d \left(\mathbb{E}\left[\|\Delta\|_2^8\right] \mathbb{E}\left[\left|\left|\left|\nabla^2 \mathsf{G}_j(\theta^*)\right|\right|\right|_2^4\right]\right)^{1/2}. \end{aligned}$$

Applying Lemma 8 yields that $\mathbb{E}[\|\Delta\|_2^8] = O(G^8/(\lambda^2 n)^4)$. Introducing the shorthand notation $g(\cdot;x) := \nabla f(\cdot;x) - \nabla F_0(\cdot)$, we can write

$$\nabla^2 G_j(\theta^*) = \frac{1}{n}\sum_{i=1}^n \nabla^2 g_j(\theta^*;X_i)$$

For every coordinate j, the random matrices $\nabla^2 g_j(\theta^*;X_i)$ $(i = 1,\ldots,n)$ are i.i.d. and mean zero. By Assumption 3, we have $\left|\left|\left|\nabla^2 g_j(\theta^*;X_i)\right|\right|\right|_2 \le 2L(X_i)$, whence we have $\mathbb{E}[\left|\left|\left|\nabla^2 g_j(\theta^*;X_i)\right|\right|\right|_2^8] \le 2^8 L^8$. Applying Lemma 16, we obtain

$$\mathbb{E}\left[\left|\left|\left|\nabla^2 G_j(\theta^*)\right|\right|\right|_2^4\right] \le O(1)L^4 n^{-2}\log^2(d),$$

and hence

$$\mathbb{E}\left[\left\|(\nabla^3 F_1(\theta^*) - \nabla^3 F_0(\theta^*))(\Delta\otimes\Delta)\right\|_2^2\right] \le O(1)\frac{G^4L^2}{\lambda^4} d\log(d) n^{-3},$$

which implies the desired result (54). From now on, terms of the form $\mathcal{R}_3$ will have no larger constants than those in the equality (54), so we ignore them.

Now we claim that

$$1_{(\mathcal{E})}\nabla^3 F_1(\theta^*)(\Delta\otimes\Delta) = \nabla^3 F_1(\theta^*)((\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}\nabla F_1(\theta^*))) + \mathcal{R}_3. \tag{55}$$

Indeed, applying the expansion (23) to the difference $\Delta = \theta_1 - \theta^*$, we have on $\mathcal{E}$ that

$$\begin{aligned}\Delta\otimes\Delta = {} & (\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}\nabla F_1(\theta^*)) + (\Sigma^{-1}P\Delta)\otimes(\Sigma^{-1}P\Delta) \\ & - (\Sigma^{-1}P\Delta)\otimes(\Sigma^{-1}\nabla F_1(\theta^*)) - (\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}P\Delta).\end{aligned}$$

We can bound each of the second three outer products in the equality above similarly; we focus on the last for simplicity. Applying the Cauchy-Schwarz inequality, we have

$$\mathbb{E}\left[\left|\left|\left|(\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}P\Delta)\right|\right|\right|_2^2\right] \le \left(\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^4\right]\mathbb{E}\left[\left\|\Sigma^{-1}P(\theta_1-\theta^*)\right\|_2^4\right]\right)^{\frac{1}{2}}.$$

From Lemmas 8 and 9, we obtain that

$$\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^4\right] = O(n^{-2}) \quad \text{and} \quad \mathbb{E}\left[\left\|\Sigma^{-1}P(\theta_1-\theta^*)\right\|_2^4\right] = O(n^{-4})$$

after an additional application of Cauchy-Schwarz for the second expectation. This shows that

$$(\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}P\Delta) = \mathcal{R}_3,$$

and a similar proof applies to the other three terms in the outer product $\Delta\otimes\Delta$. Using the linearity of $\nabla^3 F_1(\theta^*)$, we see that to prove the equality (55), all that is required is that

$$1_{(\mathcal{E}^c)}\nabla^3 F_1(\theta^*)\left((\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}\nabla F_1(\theta^*))\right) = \mathcal{R}_3. \tag{56}$$

For this, we apply Hölder's inequality several times. Indeed, we have

$$\begin{aligned}
&\mathbb{E}\left[\left\|1_{(\mathcal{E}^c)}\nabla^3 F_1(\theta^*)\left((\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}\nabla F_1(\theta^*))\right)\right\|_2^2\right] \\
&\leq \mathbb{E}[1_{(\mathcal{E}^c)}]^{1/4}\mathbb{E}\left[\left\|\nabla^3 F_1(\theta^*)\left((\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}\nabla F_1(\theta^*))\right)\right\|_2^{8/3}\right]^{3/4} \\
&\leq \mathbb{E}[1_{(\mathcal{E}^c)}]^{1/4}\mathbb{E}\left[\left|\left|\left|\nabla^3 F_1(\theta^*)\right|\right|\right|^{8/3}\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^{16/3}\right]^{3/4} \\
&\leq \mathbb{E}[1_{(\mathcal{E}^c)}]^{1/4}\mathbb{E}\left[\left|\left|\left|\nabla^3 F_1(\theta^*)\right|\right|\right|^{8}\right]^{1/4}\mathbb{E}\left[\left\|\Sigma^{-1}\nabla F_1(\theta^*)\right\|_2^{8}\right]^{2/4} = O(n^{-1}\cdot L^2\cdot n^{-2}).
\end{aligned}$$

For the final asymptotic bound, we used Equation (26) to bound $\mathbb{E}[1_{(\mathcal{E}^c)}]$, used the fact (from Assumption 3) that $\mathbb{E}[L(X)^8] \leq L^8$ to bound the term involving $\nabla^3 F_1(\theta^*)$, and applied Lemma 7 to control $\mathbb{E}[\|\Sigma^{-1}\nabla F_1(\theta^*)\|_2^8]$. Thus the equality (56) holds, and this completes the proof of the equality (55).

For the final step in the lemma, we claim that

$$-1_{(\mathcal{E})}\Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Delta = \Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_1(\theta^*) + \mathcal{R}_3. \tag{57}$$

To prove (57) requires an argument completely parallel to that for our claim (55). As before, we use the expansion (23) of the difference Δ to obtain that on $\mathcal{E}$,

$$\begin{aligned}
&-\Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Delta \\
&= \Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_1(\theta^*) - \Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Sigma^{-1}P\Delta.
\end{aligned}$$

Now apply Lemmas 8 and 9 to the final term after a few applications of Hölder's inequality. To finish the equality (57), we argue that $1_{(\mathcal{E}^c)}\Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_1(\theta^*) = \mathcal{R}_3$, which follows exactly the line of reasoning used to prove the remainder (56).

Applying equalities (54), (55), and (57) to our earlier expansion (53) yields that

$$\begin{aligned}
\Delta &= 1_{(\mathcal{E})}\big[-\Sigma^{-1}\nabla F_1(\theta^*) - \Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Delta - \Sigma^{-1}\nabla^3 F_1(\theta^*)(\Delta\otimes\Delta) \\
&\qquad + \Sigma^{-1}(\nabla^3 F_0(\theta^*) - \nabla^3 F_1(\theta'))(\Delta\otimes\Delta)\big] + 1_{(\mathcal{E}^c)}\Delta \\
&= -\Sigma^{-1}\nabla F_1(\theta^*) + \Sigma^{-1}(\nabla^2 F_1(\theta^*)-\Sigma)\Sigma^{-1}\nabla F_1(\theta^*) \\
&\qquad - \Sigma^{-1}\nabla^3 F_1(\theta^*)\left((\Sigma^{-1}\nabla F_1(\theta^*))\otimes(\Sigma^{-1}\nabla F_1(\theta^*))\right) + \mathcal{R}_3 + 1_{(\mathcal{E}^c)}\Delta.
\end{aligned}$$

Finally, the bound (26) implies that $\mathbb{E}[1_{(\mathcal{E}^c)}\|\Delta\|_2^2] \leq \mathbb{P}(\mathcal{E}^c)R^2 = O(n^{-4})$, which yields the claim.

References

A. Agarwal and J. C. Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems 24*, 2011.

A. Agarwal, P. L. Bartlett, P. Ravikumar, and M. J. Wainwright. Information-theoretic lower bounds on the oracle complexity of convex optimization. *IEEE Transactions on Information Theory*, 58 (5):3235–3249, May 2012.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011.

R. Chen, A. Gittens, and J. A. Tropp. The masked sample covariance estimator: an analysis using matrix concentration inequalities. *Information and Inference*, to appear, 2012.

A. de Acosta. Inequalities for b-valued random vectors with applications to the strong law of large numbers. *The Annals of Probability*, 9:157–161, 1981.

O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13:165–202, 2012.

J. C. Duchi, A. Agarwal, and M. J. Wainwright. Dual averaging for distributed optimization: convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57(3):592–606, 2012a.

J. C. Duchi, P. L. Bartlett, and M. J. Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012b.

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, 1993.

P. Hall. *The Bootstrap and Edgeworth Expansion*. Springer, 1992.

E. Hazan, A. Kalai, S. Kale, and A. Agarwal. Logarithmic regret algorithms for online convex optimization. In *Proceedings of the Nineteenth Annual Conference on Computational Learning Theory*, 2006.

B. Johansson, M. Rabi, and M. Johansson. A randomized incremental subgradient method for distributed optimization in networked systems. *SIAM Journal on Optimization*, 20(3):1157–1170, 2009.

R. W. Keener. *Theoretical Statistics: Topics for a Core Course*. Springer, 2010.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces*. Springer, 1991.

E. L. Lehmann and G. Casella. *Theory of Point Estimation, Second Edition*. Springer, 1998.

G. Mann, R. McDonald, M. Mohri, N. Silberman, and D. Walker. Efficient large-scale distributed training of conditional maximum entropy models. In *Advances in Neural Information Processing Systems 22*, pages 1231–1239, 2009.

C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

R. McDonald, K. Hall, and G. Mann. Distributed training strategies for the structured perceptron. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2010.

A. Nedić and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54:48–61, 2009.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

D. N. Politis, J. P. Romano, and M. Wolf. *Subsampling*. Springer, 1999.

B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

S. Rakhlin, O. Shamir, and K. Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

S. S. Ram, A. Nedić, and V. V. Veeravalli. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of Optimization Theory and Applications*, 147(3):516–545, 2010.

B. Recht, C. Re, S. Wright, and F. Niu. Hogwild: a lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems 24*, 2011.

H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *arXiv preprint arXiv:1209.1873*, 2012.

G. Sun. KDD cup track 2 `soso.com` ads prediction challenge, 2012. URL `http://www.kddcup2012.org/c/kddcup2012-track2`. Accessed August 1, 2012.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge series in statistical and probabilistic mathematics. Cambridge University Press, 1998. ISBN 0-521-49603-9.

Y. Zhang, J. C. Duchi, and M. J. Wainwright. Divide and conquer kernel ridge regression. In *Proceedings of the Twenty Sixth Annual Conference on Computational Learning Theory*, Princeton, NJ, July 2013.

M. A. Zinkevich, A. Smola, M. Weimer, and L. Li. Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems 23*, 2010.

Journal of Machine Learning Research 14 (2013) 3365-3383 Submitted 7/12; Revised 6/13; Published 11/13

PC Algorithm for Nonparanormal Graphical Models

Naftali Harris NAFTALI@STANFORD.EDU
Department of Statistics
Stanford University
Stanford, CA, USA 94305

Mathias Drton MD5@UW.EDU
Department of Statistics
University of Washington
Seattle, WA, USA 98195

Editor: Chris Meek

Abstract

The PC algorithm uses conditional independence tests for model selection in graphical modeling with acyclic directed graphs. In Gaussian models, tests of conditional independence are typically based on Pearson correlations, and high-dimensional consistency results have been obtained for the PC algorithm in this setting. Analyzing the error propagation from marginal to partial correlations, we prove that high-dimensional consistency carries over to a broader class of Gaussian copula or *nonparanormal* models when using rank-based measures of correlation. For graph sequences with bounded degree, our consistency result is as strong as prior Gaussian results. In simulations, the 'Rank PC' algorithm works as well as the 'Pearson PC' algorithm for normal data and considerably better for non-normal data, all the while incurring a negligible increase of computation time. While our interest is in the PC algorithm, the presented analysis of error propagation could be applied to other algorithms that test the vanishing of low-order partial correlations.

Keywords: Gaussian copula, graphical model, model selection, multivariate normal distribution, nonparanormal distribution

1. Introduction

Let $G = (V,E)$ be an acyclic digraph with finite vertex set, and let $X = (X_v)_{v\in V}$ be a random vector whose entries are in correspondence with the graph's vertices. Then the graph G determines a statistical model for the joint distribution of X by imposing conditional independences that can be read off from G using the concept of d-separation. These independences are natural if the edges in E encode causal/functional relationships among the random variables X_v, and a distribution that satisfies them is said to be *Markov* with respect to G. Appendix B contains a brief review of these and other key notions that are relevant to this paper. More detailed introductions to statistical modeling with directed graphs can be found in Lauritzen (1996), Pearl (2009), Spirtes et al. (2000) or Drton et al. (2009, Chapter 3). As common in the field, we use the abbreviation DAG (for 'directed acyclic graph') when referring to acyclic digraphs.

We will be concerned with the consistency of the PC algorithm, which is named for its inventors, the first two authors of Spirtes et al. (2000). This algorithm uses conditional independence tests to infer a DAG from data. Alongside greedy-search techniques that optimize information criteria, the PC algorithm is one of the main methods for inference of directed graphs. Recent applications of the

PC algorithm can be found in Maathuis et al. (2010), Schmidberger et al. (2011), Le et al. (2013), and Verdugo et al. (2013).

Graph inference is complicated by the fact that two DAGs $G = (V,E)$ and $H = (V,F)$ with the same vertex set V may be *Markov equivalent*, that is, they may possess the same d-separation relations and, consequently, induce the same statistical model. Hence, the goal becomes estimation of the Markov equivalence class of an acyclic digraph G. For representation of the equivalence class, prior work considers a particular partially directed graph $C(G)$, for which it holds that $C(G) = C(H)$ if and only if the two DAGs G and H are Markov equivalent; see Andersson et al. (1997) and Chickering (2002). The graph $C(G)$ may contain both directed and undirected edges, and it is acyclic in the sense of its directed subgraph having no directed cycles. We will refer to $C(G)$ as the *completed partially directed acyclic graph* (CPDAG), but other terminology such as the *essential graph* is in use.

The PC algorithm uses conditional independence tests to infer a CPDAG from data (Spirtes et al., 2000). In its population version, the algorithm amounts to a clever scheme to reconstruct the CPDAG $C(G)$ from answers to queries about d-separation relations in the underlying DAG G. Theorem 1 summarizes the properties of the PC algorithm that are relevant for the present paper. For a proof of the theorem as well as a compact description of the PC algorithm we refer the reader to Kalisch and Bühlmann (2007). Recall that the degree of a node is the number of edges it is incident to, and that the degree of a DAG G is the maximum degree of any node, which we denote by $\deg(G)$.

Theorem 1 *Given only the ability to check d-separation relations in a DAG G, the PC algorithm finds the CPDAG $C(G)$ by checking whether pairs of distinct nodes are d-separated by sets S of cardinality $|S| \leq \deg(G)$.*

Let X_A denote the subvector $(X_v)_{v\in A}$. The joint distribution of a random vector $X = (X_v)_{v\in V}$ is *faithful* to a DAG G if, for any triple of pairwise disjoint subsets $A,B,S \subset V$, we have that S d-separates A and B in G if and only if X_A and X_B are conditionally independent given X_S; it is customary to denote this conditional independence by $X_A \perp\!\!\!\perp X_B \,|\, X_S$. Under faithfulness, statistical tests of conditional independence can be used to determine d-separation relations in a DAG and lead to a sample version of the PC algorithm that is applicable to data.

If X follows the multivariate normal distribution $N(\mu,\Sigma)$, with positive definite covariance matrix Σ, then

$$X_A \perp\!\!\!\perp X_B \,|\, X_S \quad \Longleftrightarrow \quad X_u \perp\!\!\!\perp X_v \,|\, X_S \quad \forall\, u \in A,\, v \in B.$$

Moreover, the pairwise conditional independence of X_u and X_v given X_S is equivalent to the vanishing of the *partial correlation* $\rho_{uv|S}$, that is, the correlation obtained from the bivariate normal conditional distribution of (X_u, X_v) given X_S. The iterations of the PC algorithm make use of the recursion

$$\rho_{uv|S} = \frac{\rho_{uv|S\setminus w} - \rho_{uw|S\setminus w}\rho_{vw|S\setminus w}}{\sqrt{\left(1-\rho^2_{uw|S\setminus w}\right)\left(1-\rho^2_{vw|S\setminus w}\right)}}, \tag{1}$$

for any $w \in S$, where $\rho_{uv|\emptyset} = \rho_{uv}$ is the correlation of u and v. Our later theoretical analysis will use the fact that

$$\rho_{uv|S} = -\frac{\Psi^{-1}_{uv}}{\sqrt{\Psi^{-1}_{uu}\Psi^{-1}_{vv}}}, \tag{2}$$

where $\Psi = \Sigma_{(u,v,S),(u,v,S)}$ is the concerned principal submatrix of Σ.

A natural estimate of $\rho_{uv|S}$ is the sample partial correlation obtained by replacing Σ with the empirical covariance matrix of available observations. Sample partial correlations derived from independent normal observations have favorable distributional properties (Anderson, 2003, Chapter 4), which form the basis for the work of Kalisch and Bühlmann (2007) who treat the PC algorithm in the Gaussian context with conditional independence tests based on sample partial correlations. The main results in Kalisch and Bühlmann (2007) show high-dimensional consistency of the PC algorithm, when the observations form a sample of independent normal random vectors that are faithful to a suitably sparse DAG.

The purpose of this paper is to show that the PC algorithm has high-dimensional consistency properties for a broader class of distributions, when standard Pearson-type empirical correlations are replaced by rank-based measures of correlations in tests of conditional independence. The broader class we consider includes continuous distributions with Gaussian copula. Phrased in the terminology of Liu et al. (2009), we consider *nonparanormal* distributions. Recall that a correlation matrix is a covariance matrix with all diagonal entries equal to one.

Definition 2 *Let $f = (f_v)_{v\in V}$ be a collection of strictly increasing functions $f_v : \mathbb{R} \to \mathbb{R}$, and let $\Sigma \in \mathbb{R}^{V\times V}$ be a positive definite correlation matrix. The nonparanormal distribution $NPN(f,\Sigma)$ is the distribution of the random vector $(f_v(Z_v))_{v\in V}$ for $(Z_v)_{v\in V} \sim N(0,\Sigma)$.*

Taking the functions f_v to be affine shows that all multivariate normal distributions are also nonparanormal. If $X \sim NPN(f,\Sigma)$, then the univariate marginal distribution for a coordinate, say X_v, may have any continuous cumulative distribution function F, as we may take $f_v = F^- \circ \Phi$, where Φ is the standard normal distribution function and $F^-(u) = \inf\{x : F(x) \geq u\}$. Note that f_v need not be continuous.

Definition 3 *The nonparanormal graphical model $NPN(G)$ associated with a DAG G is the set of all distributions $NPN(f,\Sigma)$ that are Markov with respect to G.*

Since the marginal transformations f_v are deterministic, the dependence structure in a nonparanormal distribution corresponds to that in the underlying latent multivariate normal distribution. In other words, if $X \sim NPN(f,\Sigma)$ and $Z \sim N(0,\Sigma)$, then it holds for any triple of pairwise disjoint sets $A,B,S \subset V$ that

$$X_A \perp\!\!\!\perp X_B \,|\, X_S \iff Z_A \perp\!\!\!\perp Z_B \,|\, Z_S.$$

Hence, for two nodes u and v and a separating set $S \subset V \setminus \{u,v\}$, it holds that

$$X_u \perp\!\!\!\perp X_v \,|\, X_S \iff \rho_{uv|S} = 0, \tag{3}$$

with $\rho_{uv|S}$ calculated from Σ as in (1) or (2). In light of this equivalence, we will occasionally speak of a correlation matrix Σ being Markov or faithful to a DAG, meaning that the requirement holds for any distribution $NPN(f,\Sigma)$.

In the remainder of the paper we study the PC algorithm in the nonparanormal context, proposing the use of Spearman's rank correlation and Kendall's τ for estimation of the correlation matrix parameter of a nonparanormal distribution. In Section 2, we review how transformations of Spearman's rank correlation and Kendall's τ yield accurate estimators of the latent Gaussian correlations. In particular, we summarize tail bounds from Liu et al. (2012a). Theorem 8 in Section 4 gives

our main result, an error bound for the output of the PC algorithm when correlations are used to determine nonparanormal conditional independence. In Corollary 9, we describe high-dimensional asymptotic scenarios and suitable conditions that lead to consistency of the PC algorithm. The proof of Theorem 8 is given in Section 3, which provides an analysis of error propagation from marginal to partial correlations. Our numerical work in Section 5 makes a strong case for the use of rank correlations in the PC algorithm. Some concluding remarks are given in Section 6.

2. Rank Correlations

Let (X,Y) be a pair of random variables, and let F and G be the cumulative distribution functions of X and Y, respectively. Spearman's ρ for the bivariate distribution of (X,Y) is defined as

$$\rho^S = \mathrm{Corr}\left(F(X), G(Y)\right),$$

that is, it is the ordinary Pearson correlation between the quantiles $F(X)$ and $G(Y)$. Another classical measure of correlation is Kendall's τ, defined as

$$\tau = \mathrm{Corr}\left(\mathrm{sign}\left(X - X'\right), \mathrm{sign}\left(Y - Y'\right)\right)$$

where (X',Y') is an independent copy of (X,Y), that is, (X',Y') and (X,Y) are independent and have the same distribution.

Suppose $(X_1,Y_1),\ldots(X_n,Y_n)$ are independent pairs of random variables, each pair distributed as (X,Y). Let $\mathrm{rank}(X_i)$ be the rank of X_i among $X_1,\ldots,X_n$. In the nonparanormal setting, the marginal distributions are continuous so that ties occur with probability zero, making ranks well-defined. The natural estimator of ρ^S is the sample correlation among ranks, that is,

$$\begin{aligned}\hat{\rho}^S &= \frac{\frac{1}{n}\sum_{i=1}^n \left(\frac{\mathrm{rank}(X_i)}{n+1} - \frac{1}{2}\right)\left(\frac{\mathrm{rank}(Y_i)}{n+1} - \frac{1}{2}\right)}{\sqrt{\frac{1}{n}\sum_{i=1}^n \left(\frac{\mathrm{rank}(X_i)}{n+1} - \frac{1}{2}\right)^2}\sqrt{\frac{1}{n}\sum_{i=1}^n \left(\frac{\mathrm{rank}(Y_i)}{n+1} - \frac{1}{2}\right)^2}} \\ &= 1 - \frac{6}{n\left(n^2-1\right)}\sum_{i=1}^n \left(\mathrm{rank}(X_i) - \mathrm{rank}(Y_i)\right)^2,\end{aligned}$$

which can be computed in $O(n\log n)$ time. Kendall's τ may be estimated by

$$\hat{\tau} = \frac{2}{n(n-1)}\sum_{1\le i<j\le n} \mathrm{sign}\left(X_i - X_j\right)\mathrm{sign}\left(Y_i - Y_j\right).$$

A clever algorithm using sorting and binary trees to compute $\hat{\tau}$ in time $O(n\log n)$ instead of the naive $O(n^2)$ time has been developed by Christensen (2005).

It turns out that simple trigonometric transformations of $\hat{\rho}^S$ and $\hat{\tau}$ are excellent estimators of the population Pearson correlation for multivariate normal data. In particular, Liu et al. (2012a) show that if (X,Y) are bivariate normal with $\mathrm{Corr}(X,Y) = \rho$, then

$$\mathbb{P}\left(\left|2\sin\left(\frac{\pi}{6}\hat{\rho}^S\right) - \rho\right| > \varepsilon\right) \;\le\; 2\exp\left(-\frac{2}{9\pi^2}n\varepsilon^2\right) \tag{4}$$

and

$$\mathbb{P}\left(\left|\sin\left(\frac{\pi}{2}\hat{\tau}\right) - \rho\right| > \varepsilon\right) \;\le\; 2\exp\left(-\frac{2}{\pi^2}n\varepsilon^2\right). \tag{5}$$

Clearly, $\hat{\rho}^S$ and $\hat{\tau}$ depend on the observations $(X_1,Y_1),\ldots(X_n,Y_n)$ only through their ranks. Since ranks are preserved under strictly increasing functions, (4) and (5) still hold if $(X,Y) \sim NPN(f,\Sigma)$ with Pearson correlation $\rho = \Sigma_{xy}$ in the underlying latent bivariate normal distribution. Throughout the rest of this paper, we will assume that we have some estimator $\hat{\rho}$ of ρ which has the property that, for nonparanormal data,

$$\mathbb{P}(|\hat{\rho}-\rho| > \varepsilon) \;<\; A\exp\left(-Bn\varepsilon^2\right) \tag{6}$$

for fixed constants $0 < A, B < \infty$. As just argued, the estimators considered in (4) and (5) both have this property.

When presented with multivariate observations from a distribution $NPN(f,\Sigma)$, we apply the estimator from (6) to every pair of coordinates to obtain an estimator $\hat{\Sigma}$ of the correlation matrix parameter. Plugging $\hat{\Sigma}$ into (1) or equivalently into (2) gives partial correlation estimators that we denote $\hat{\rho}_{uv|S}$.

3. Error Propagation from Marginal to Partial Correlations

The PC algorithm leverages statistical decisions on conditional independence. An analysis of the algorithm in the context of nonparanormal distributions thus requires bounds on errors in partial correlations. The following Lemma 4 is our main tool. It provides a uniform bound on errors in partial correlations when a uniform bound on errors in marginal correlations is available. At times we will write such uniform bounds in terms of the l_∞ vector norm of a matrix. For matrix $A = (a_{ij}) \in \mathbb{R}^{q\times q}$ we denote this norm by

$$\|A\|_\infty = \max_{1\le i,j\le q} |a_{ij}|.$$

Some proofs involve the spectral norm $\|A\|$, that is, the square-root of the maximal eigenvalue of A^TA.

Lemma 4 (Errors in partial correlations) *Suppose $\Sigma \in \mathbb{R}^{q\times q}$ is a positive definite matrix with minimal eigenvalue $\lambda_{\min} > 0$. If $\hat{\Sigma} \in \mathbb{R}^{q\times q}$ satisfies*

$$\|\hat{\Sigma}-\Sigma\|_\infty < \frac{c\lambda_{\min}^2}{(2+c)q + \lambda_{\min} cq}$$

with $c > 0$, then all partial correlations are well-defined and their differences are bounded as

$$|\hat{\rho}_{uv|\setminus\{u,v\}} - \rho_{uv|\setminus\{u,v\}}| \;:=\; \left|\frac{\Sigma^{-1}_{uv}}{\sqrt{\Sigma^{-1}_{uu}\Sigma^{-1}_{vv}}} - \frac{\hat{\Sigma}^{-1}_{uv}}{\sqrt{\hat{\Sigma}^{-1}_{uu}\hat{\Sigma}^{-1}_{vv}}}\right| < c, \quad 1 \le u < v \le q.$$

The proof of Lemma 4 follows by combining the conclusions of Lemmas 5, 6 and 7 from this section. The first of these, that is, Lemma 5, invokes classical results on error propagation in matrix inversion.

Lemma 5 (Matrix inversion) *Suppose $\Sigma \in \mathbb{R}^{q\times q}$ is a positive definite matrix with minimal eigenvalue $\lambda_{\min} > 0$. If $E \in \mathbb{R}^{q\times q}$ is a matrix of errors with $\|E\|_\infty < \varepsilon < \lambda_{\min}/q$, then $\Sigma + E$ is invertible and*

$$\|(\Sigma+E)^{-1} - \Sigma^{-1}\|_\infty \le \frac{q\varepsilon/\lambda_{\min}^2}{1 - q\varepsilon/\lambda_{\min}}.$$

Proof First, note that

$$\|E\|_\infty \leq \|E\| \leq q\|E\|_\infty; \tag{7}$$

see entries $(2,6)$ and $(6,2)$ in the table on p. 314 in Horn and Johnson (1990). Using the submultiplicativity of a matrix norm, the second inequality in (7), and our assumption on ε, we find that

$$\|E\Sigma^{-1}\| \leq \|\Sigma^{-1}\| \cdot \|E\| < \frac{q\varepsilon}{\lambda_{\min}} < 1. \tag{8}$$

As discussed in Horn and Johnson (1990, Section 5.8), this implies that $I + E\Sigma^{-1}$ and thus also $\Sigma + E$ is invertible. Moreover, by the first inequality in (7) and inequality (5.8.2) in Horn and Johnson (1990), we obtain that

$$\|(\Sigma+E)^{-1} - \Sigma^{-1}\|_\infty \leq \|(\Sigma+E)^{-1} - \Sigma^{-1}\| \leq \|\Sigma^{-1}\| \cdot \frac{\|E\Sigma^{-1}\|}{1 - \|E\Sigma^{-1}\|}.$$

Since the function $x \mapsto x/(1-x)$ is increasing for $x < 1$, our claim follows from the fact that $\|\Sigma^{-1}\| = 1/\lambda_{\min}$ and the inequality $\|E\Sigma^{-1}\| < q\varepsilon/\lambda_{\min}$ from (8). ■

Lemma 6 (Diagonal of inverted correlation matrix) *If $\Sigma \in \mathbb{R}^{q\times q}$ is a positive definite correlation matrix, then the diagonal entries of $\Sigma^{-1} = (\sigma^{ij})$ satisfy $\sigma^{ii} \geq 1$.*

Proof The claim is trivial for $q = 1$. So assume $q \geq 2$. By symmetry, it suffices to consider the entry σ^{qq}, and we partition the matrix as

$$\Sigma = \begin{pmatrix} A & b \\ b^T & 1 \end{pmatrix}$$

with $A \in \mathbb{R}^{(q-1)\times(q-1)}$ and $b \in \mathbb{R}^{q-1}$. By the Schur complement formula for the inverse of a partitioned matrix,

$$\sigma^{qq} = \frac{1}{1 - b^T A^{-1} b};$$

compare Horn and Johnson (1990, Section 0.7.3). Since A is positive definite, so is A^{-1}. Hence, $b^T A^{-1} b \geq 0$. Since Σ^{-1} is positive definite, σ^{qq} cannot be negative, and so we deduce that $\sigma_{qq} \geq 1$, with equality if and only if $b = 0$. ■

The next lemma addresses the error propagation from the inverse of a correlation matrix to partial correlations.

Lemma 7 (Correlations) *Let $A = (a_{ij})$ and $B = (b_{ij})$ be symmetric 2×2 matrices. If A is positive definite with $a_{11}, a_{22} \geq 1$ and $\|A - B\|_\infty < \delta < 1$, then*

$$\left| \frac{a_{12}}{\sqrt{a_{11}a_{22}}} - \frac{b_{12}}{\sqrt{b_{11}b_{22}}} \right| < \frac{2\delta}{1-\delta}.$$

Proof Without loss of generality, suppose $a_{12} \geq 0$. Since $\|A-B\|_\infty < \delta$,

$$\frac{b_{12}}{\sqrt{b_{11}b_{22}}} - \frac{a_{12}}{\sqrt{a_{11}a_{22}}} < \frac{a_{12}+\delta}{\sqrt{(a_{11}-\delta)(a_{22}-\delta)}} - \frac{a_{12}}{\sqrt{a_{11}a_{22}}}$$
$$= \frac{\delta}{\sqrt{(a_{11}-\delta)(a_{22}-\delta)}} + a_{12}\left(\frac{1}{\sqrt{(a_{11}-\delta)(a_{22}-\delta)}} - \frac{1}{\sqrt{a_{11}a_{22}}}\right).$$

Using that $a_{11}, a_{22} \geq 1$ to bound the first term and $a_{12}^2 < a_{11}a_{22}$ to bound the second term, we obtain that

$$\frac{b_{12}}{\sqrt{b_{11}b_{22}}} - \frac{a_{12}}{\sqrt{a_{11}a_{22}}} < \frac{\delta}{1-\delta} + \sqrt{a_{11}a_{22}}\left(\frac{1}{\sqrt{(a_{11}-\delta)(a_{22}-\delta)}} - \frac{1}{\sqrt{a_{11}a_{22}}}\right)$$
$$= \frac{\delta}{1-\delta} + \left(\sqrt{\frac{a_{11}}{a_{11}-\delta}\cdot\frac{a_{22}}{a_{22}-\delta}} - 1\right).$$

Since the function $x \mapsto x/(x-\delta)$ is decreasing, we may use our assumption that $a_{11}, a_{22} \geq 1$ to get the bound

$$\frac{b_{12}}{\sqrt{b_{11}b_{22}}} - \frac{a_{12}}{\sqrt{a_{11}a_{22}}} < \frac{\delta}{1-\delta} + \left(\sqrt{\frac{1}{1-\delta}\cdot\frac{1}{1-\delta}} - 1\right) = \frac{2\delta}{1-\delta}$$

A similar argument yields that

$$\frac{a_{12}}{\sqrt{a_{11}a_{22}}} - \frac{b_{12}}{\sqrt{b_{11}b_{22}}} < \frac{2\delta}{1+\delta},$$

from which our claim follows. ■

4. Rank PC Algorithm

Based on the equivalence (3), we may use the rank-based partial correlation estimates $\hat{\rho}_{uv|S}$ to test conditional independences. In other words, we conclude that

$$X_u \perp\!\!\!\perp X_v | X_S \quad \Longleftrightarrow \quad \left|\hat{\rho}_{uv|S}\right| \leq \gamma, \tag{9}$$

where $\gamma \in [0,1]$ is a fixed threshold. We will refer to the PC algorithm that uses the conditional independence tests from (9) as the 'Rank PC' (RPC) algorithm. We write $\hat{C}_\gamma(G)$ for the output of the RPC algorithm with tuning parameter γ.

The RPC algorithm consist of two parts. The first part computes the correlation matrix $\hat{\Sigma} = (\hat{\rho}_{uv})$ in time $O(p^2 n \log n)$, where $p := |V|$. This computation takes $O(\log n)$ longer than its analogue under use of Pearson correlations. The second part of the algorithm is independent of the type of correlations involved. It determines partial correlations and performs graphical operations. For an accurate enough estimate of a correlation matrix Σ that is faithful to a DAG G, this second part takes $O(p^{\deg(G)})$ time in the worst case, but it is often much faster; compare Kalisch and Bühlmann (2007). For high-dimensional data with n smaller than p, the computation time for RPC is dominated by the second part, the PC-algorithm component. Moreover, in practice, one may wish to apply RPC

for several different values of γ, in which case the estimate $\hat{\Sigma}$ needs to be calculated only once. As a result, Rank PC takes only marginally longer to compute than Pearson PC for high-dimensional data.

What follows is our main result about the correctness of RPC. For a correlation matrix $\Sigma \in \mathbb{R}^{V\times V}$, let

$$c_{\min}(\Sigma) := \min\left\{|\rho_{uv|S}| : u,v \in V, S \subseteq V\setminus\{u,v\}, \rho_{uv|S} \neq 0\right\} \tag{10}$$

be the minimal magnitude of any non-zero partial correlation, and let $\lambda_{\min}(\Sigma)$ be the minimal eigenvalue. Then for any integer $q \geq 2$, let

$$c_{\min}(\Sigma, q) := \min\left\{c_{\min}(\Sigma_{I,I}) : I \subseteq V, |I| \leq q\right\}, \quad \text{and} \tag{11}$$

$$\lambda_{\min}(\Sigma, q) := \min\left\{\lambda_{\min}(\Sigma_{I,I}) : I \subseteq V, |I| \leq q\right\} \tag{12}$$

be the minimal magnitude of a non-zero partial correlation and, respectively, the minimal eigenvalue of any principal submatrix of order at most q.

Theorem 8 (Error bound for RPC-algorithm) *Let $X_1,\ldots,X_n$ be a sample of independent observations drawn from a nonparanormal distribution $NPN(f,\Sigma)$ that is faithful to a DAG G with p nodes. For $q := \deg(G)+2$, let $c := c_{\min}(\Sigma,q)$ and $\lambda := \lambda_{\min}(\Sigma,q)$. If $n > q$, then there exists a threshold $\gamma \in [0,1]$ for which*

$$\mathbb{P}\big(\hat{C}_\gamma(G) \neq C(G)\big) \;\leq\; \frac{A}{2}p^2 \exp\left(-\frac{B\lambda^4 n c^2}{36q^2}\right),$$

where $0 < A,B < \infty$ are the constants from (6).

We remark that while all subsets of size q appear in the definitions in (11) and (12), our proof of Theorem 8 only requires the corresponding minima over those principal submatrices that are actually inverted in the run of the PC-algorithm.

Proof (Theorem 8) We will show that our claimed probability bound for the event $\hat{C}_\gamma(G) \neq C(G)$ holds when the threshold in the RPC algorithm is $\gamma = c/2$. By Theorem 1, if all conditional independence tests for conditioning sets of size $|S| \leq \deg(G)$ make correct decisions, then the output of the RPC algorithm $\hat{C}_\gamma(G)$ is equal to the CPDAG $C(G)$. When $\gamma = c/2$, the conditional independence test accepts a hypothesis $X_u \perp\!\!\!\perp X_v | X_S$ if and only if $|\hat{\rho}_{uv|S}| < \gamma = c/2$. Hence, the test makes a correct decision if $|\hat{\rho}_{uv|S} - \rho_{uv|S}| < c/2$ because all non-zero partial correlations for $|S| \leq \deg(G)$ are bounded away from zero by c; recall (10) and (11). It remains to argue, using the error analysis from Lemma 4, that the event $|\hat{\rho}_{uv|S} - \rho_{uv|S}| \geq c/2$ occurs with small enough probability when $|S| \leq \deg(G)$.

Suppose our correlation matrix estimate $\hat{\Sigma} = (\hat{\rho}_{uv})$ satisfies $\|\hat{\Sigma} - \Sigma\|_\infty < \varepsilon$ for

$$\varepsilon = \frac{c\lambda^2}{(4+c)q + \lambda c q} = \frac{\lambda^2 c/2}{(2+c/2)q + \lambda q c/2} > 0. \tag{13}$$

Choose any two nodes $u,v \in V$ and a set $S \subseteq V\setminus\{u,v\}$ with $|S| \leq \deg(G) = q-2$. Let $I = \{u,v\}\cup S$. Applying Lemma 4 to the $I\times I$ submatrix of Σ and $\hat{\Sigma}$ yields

$$|\hat{\rho}_{uv|S} - \rho_{uv|S}| \;<\; \frac{c}{2}.$$

Therefore, $\|\hat{\Sigma}-\Sigma\|_\infty < \varepsilon$ implies that our tests decide all conditional independences correctly in the RPC algorithm.

Next, using (6) and a union bound, we find that

$$\begin{aligned}\mathbb{P}\left(\hat{C}_\gamma(G) \neq C(G)\right) &\leq \mathbb{P}\left(|\hat{\Sigma}_{uv}-\Sigma_{uv}| \geq \varepsilon \text{ for some } u,v \in V\right) \\ &\leq A\frac{p(p-1)}{2}\exp\left(-Bn\varepsilon^2\right).\end{aligned}$$

Plugging in the definition of ε gives the claimed inequality

$$\mathbb{P}\left(\hat{C}_\gamma(G) \neq C(G)\right) \leq \frac{A}{2}p^2\exp\left(-\frac{B\lambda^4 nc^2}{((4+c)q+\lambda cq)^2}\right) \leq \frac{A}{2}p^2\exp\left(-\frac{B\lambda^4 nc^2}{36q^2}\right)$$

because $c \leq 1$ and $\lambda \leq 1$. The inequality $c \leq 1$ holds trivially because partial correlations are in $[-1,1]$. The inequality $\lambda \leq 1$ holds because a $q \times q$ correlation matrix has trace q, this trace is equal to the sum of the q eigenvalues, and λ is the minimal eigenvalue. ■

From the probability bound in Theorem 8, we may deduce high-dimensional consistency of RPC. For two positive sequences (s_n) and (t_n), we write $s_n = O(t_n)$ if $s_n \leq Mt_n$, and $s_n = \Omega(t_n)$ if $s_n \geq Mt_n$ for a constant $0 < M < \infty$.

Corollary 9 (Consistency of RPC-algorithm) *Let (G_n) be a sequence of DAGs. Let p_n be the number of nodes of G_n, and let $q_n = \deg(G_n)+2$. Suppose (Σ_n) is a sequence of $p_n \times p_n$ correlation matrices, with Σ_n faithful to G_n. Suppose further that there are constants $0 \leq a,b,d,f < 1$ that govern the growth of the graphs as*

$$\log p_n = O(n^a), \qquad\qquad q_n = O(n^b),$$

and minimal signal strengths and eigenvalues as

$$c_{\min}(\Sigma_n, q_n) = \Omega(n^{-d}), \qquad\qquad \lambda_{\min}(\Sigma_n, q_n) = \Omega(n^{-f}).$$

If $a+2b+2d+4f < 1$, then there exists a sequence of thresholds γ_n for which

$$\lim_{n\to\infty}\mathbb{P}\left(\hat{C}_{\gamma_n}(G_n) = C(G_n)\right) = 1,$$

where $\hat{C}_{\gamma_n}(G_n)$ is the output of the RPC algorithm for a sample of independent observations $X_1,\ldots,X_n$ from a nonparanormal distribution $NPN(\cdot,\Sigma_n)$.

Proof By Theorem 8, for large enough n, we can pick a threshold γ_n such that

$$\mathbb{P}(\hat{C}_{\gamma_n}(G_n) \neq C(G_n) \leq A'\exp\left(2n^a - B'n^{1-2b-2d-4f}\right)$$

for constants $0 < A', B' < \infty$. The bound goes to zero if $1-2b-2d-4f > a$. ■

As previously mentioned, Kalisch and Bühlmann (2007) prove a similar consistency result in the Gaussian case. Whereas our proof consists of propagation of errors from correlation to partial

correlation estimates, their proof appeals to Fisher's result that under Gaussianity, sample partial correlations follow the same type of distribution as sample correlations when the sample size is adjusted by subtracting the cardinality of the conditioning set (Anderson, 2003, Chapter 4). It is then natural to work with a bound on the partial correlations associated with small conditioning sets. More precisely, Kalisch and Bühlmann (2007) assume that there is a constant $0 \leq M < 1$ such that for any n, the partial correlations $\rho_{uv|S}$ of the matrix Σ_n satisfy

$$|\rho_{uv|S}| \leq M \qquad \forall\, u,v \in V,\, S \subseteq V \setminus \{u,v\},\, |S| \leq q_n. \tag{14}$$

It is then no longer necessary to involve the minimal eigenvalues from (12). The work in Kalisch and Bühlmann (2007) is thus free of an analogue to our constant f. Stated for the case of polynomial growth of p_n (with $a = 0$), their result gives consistency when $b + 2d < 1$; our constant b corresponds to $1 - b$ in Kalisch and Bühlmann (2007). The condition from Corollary 9, on the other hand, requires $2b + 2d < 1$ even if $f = 0$. This is more restrictive as larger b allows for faster growth in the degree of the graphs and larger d allows for faster decay of the minimal signal strength.

In the important special case of bounded degree, however, our nonparanormal result is just as strong as the previously established Gaussian consistency guarantee. Staying with polynomial growth of p_n, that is, $a = 0$, suppose the sequence of graph degrees $\deg(G_n)$ is indeed bounded by a fixed constant, say $q_0 - 2$. Then clearly, $b = 0$. Moreover, the set of correlation matrices of size q_0 satisfying (14) with $q_n = q_0$ is compact. Since the smallest eigenvalue is a continuous function, the infimum of all eigenvalues of such matrices is achieved for some invertible matrix. Hence, the smallest eigenvalue is bounded away from zero, and we conclude that $f = 0$. Corollary 9 thus implies consistency if $2d < 1$, or if $d < \frac{1}{2} = \frac{1-b}{2}$, precisely as in Kalisch and Bühlmann (2007). (No generality is lost by assuming $a = 0$; in either one of the compared results this constant is involved solely in a union bound over order p^2 events.)

5. Numerical Experiments

In this section we evaluate the finite-sample properties of the RPC algorithm in simulations and in an application to gene expression data. In implementations of the PC algorithm in the `pcalg` package for R (Kalisch et al., 2012) and other software such as `Tetrad IV`,[1] the Gaussian conditional independence tests use a fixed level $\alpha \in [0,1]$ and decide that

$$X_u \perp\!\!\!\perp X_v | X_S \iff \sqrt{n - |S| - 3}\left|\frac{1}{2}\log\left(\frac{1+\hat{\rho}_{uv|S}}{1-\hat{\rho}_{uv|S}}\right)\right| \leq \Phi^{-1}(1-\alpha/2)\,. \tag{15}$$

If the observations are multivariate normal and $\hat{\rho}_{uv|S}$ are sample partial correlations then α is an asymptotic significance level for the test. The sample size adjustment from n to $n - |S| - 3$ achieves a bias-correction (Anderson, 2003).

Suppose for a moment that in (15) the square root of $n - |S| - 3$ was simply $\sqrt{n}$. Then, for fixed n and α, the acceptance region in (15) could be translated into a corresponding fixed value for γ in (9). Hence, our Theorem 8 would apply directly when plugging rank correlations into the mentioned software implementations of the PC algorithm. With the sample size adjustment from n to $n - |S| - 3$, however, the value of γ depends on $|S|$ and further arguments are needed. We postpone these to Appendix A, where we show that the sample size adjustment has indeed no effect on the consistency result in Corollary 9.

1. Tetrad IV can be found at `http://www.phil.cmu.edu/projects/tetrad`.

5.1 Simulations

We compare RPC to two other versions of the PC-algorithm: (i) 'Pearson-PC', by which we mean the standard approach of using sample partial correlations to test Gaussian conditional independences, and (ii) 'Q_n-PC', which is based on a robust estimator of the covariance matrix and was considered in Kalisch and Bühlmann (2008). All our computations are done with the `pcalg` package for R.

Following Kalisch and Bühlmann (2007), we simulate random DAGs and sample from probability distributions faithful to them. Fix a sparsity parameter $s \in [0,1]$ and enumerate the vertices as $V = \{1,\dots,p\}$. Then we generate a DAG by including the edge $u \to v$ with probability s, independently for each pair (u,v) with $1 \le u < v \le p$. In this scheme, each node has the same expected degree $(p-1)s$.

Given a DAG $G = (V,E)$, let $\Lambda = (\lambda_{uv})$ be a $p \times p$ matrix with $\lambda_{uv} = 0$ if $u \to v \notin E$. Furthermore, let $\varepsilon = (\varepsilon_1,\dots,\varepsilon_p)$ be a vector of independent random variables. Then the random vector X solving the equation system

$$X = \Lambda X + \varepsilon \tag{16}$$

is well-known to be Markov with respect to G. Here, we draw the edge coefficients λ_{uv}, $u \to v \in E$, independently from a uniform distribution on the interval $(0.1,1)$. For such a random choice, with probability one, the vector X solving (16) is faithful with respect to G. We consider three different types of data:

(i) multivariate normal observations, which we generate by taking ε in (16) to have independent standard normal entries;

(ii) observations with Gaussian copula obtained by transforming the marginals of the normal random vectors from (i) to an $F_{1,1}$-distribution;

(iii) contaminated data, for which we generate the entries of ε in (16) as independent draws from a 80-20 mixture between a standard normal and a standard Cauchy distribution.

The contaminated distributions in (iii) do not belong to the nonparanormal class.

For the simulations we sample from two graph distributions: A small graph on ten vertices with an expected vertex degree of three, and a larger graph on one hundred vertices with an expected vertex degree of six. For each $n \in \{50,1000\}$ and each of the three types of data listed above, we sample 201 random graphs from both the small and large graph distributions, and then sample n observations from the graph with the given data distribution.

For each resulting combination, we run each of the three considered versions of the PC algorithm on a grid of α's ranging from 10^{-100} to 0.8. We consider the RPC algorithm in the version that uses Spearman correlations as in (4); the results for Kendall's τ were similar. For each estimated skeleton, we compute the proportions of true and of false positives by comparing the estimated skeleton to the true skeleton. The skeleton of a graph G is the undirected graph with edges between nodes that are adjacent in G. Finally, we compute the area under the receiver operating characteristic curve (AUC) for each of the 201 repetitions. Mean areas with standard deviation in parenthesis are listed in Tables 1- 3.

A clear message emerges from the tables. First, Table 1 shows that for normal data, RPC performs only marginally worse than Pearson-PC. The Q_n-PC algorithm does well on larger sample sizes, but it not as good on smaller sample sizes. Second, Table 2 shows a dramatic relative gain

	Pearson-PC	Q_n-PC	RPC
Small graph, $n = 50$	0.824 (0.065)	0.734 (0.102)	0.809 (0.072)
Small graph, $n = 1000$	0.938 (0.050)	0.930 (0.053)	0.936 (0.050)
Large graph, $n = 50$	0.721 (0.016)	0.584 (0.022)	0.706 (0.016)
Large graph, $n = 1000$	0.837 (0.023)	0.830 (0.023)	0.835 (0.023)

Table 1: Mean AUC for Normal data

	Pearson-PC	Q_n-PC	RPC
Small graph, $n = 50$	0.668 (0.079)	0.506 (0.062)	0.813 (0.067)
Small graph, $n = 1000$	0.774 (0.068)	0.566 (0.082)	0.930 (0.054)
Large graph, $n = 50$	0.587 (0.012)	0.502 (0.004)	0.704 (0.016)
Large graph, $n = 1000$	0.678 (0.021)	0.525 (0.011)	0.833 (0.024)

Table 2: Mean AUC for Nonparanormal data

in performance for RPC for the Gaussian copula data with $F_{1,1}$ marginals. As expected, the performance of RPC on nonparanormal data is the same as on normal data, while that of Pearson-PC and Q_n-PC deteriorate. Finally, Table 3 shows that RPC continues to do well in the presence of contaminated data, the mean AUC for the other two algorithms is significantly lower. Curiously, despite using a robust covariance matrix estimator, the Q_n-PC performs substantially worse than Pearson-PC on this data.

5.2 Gene Expression Data

While Kendall's τ and Spearman's rank correlation give similar results for continuous observations from a distribution with Gaussian copula, the two measures of correlation can give quite different results in applications. We illustrate this for data on gene expression in yeast from Brem and Kruglyak (2005), where we focus on $p = 54$ genes from the MAPK signaling pathway as was done in Sun and Li (2012). The sample size is $n = 112$.

When plotting histograms of the expression measurements for each of the 54 genes, the majority of the plots do not show any obvious deviation from normality but, as one might suspect, there are several with signs of skewness as well as some outliers. Moreover, for five genes, the marginal distribution appears to be bimodal; see Figure 1 for an example. Multimodal marginals can arise under nonparanormal distributions, which thus have the potential to alleviate the effects of such

	Pearson-PC	Q_n-PC	RPC
Small graph, $n = 50$	0.781 (0.075)	0.656 (0.102)	0.819 (0.073)
Small graph, $n = 1000$	0.905 (0.078)	0.859 (0.110)	0.939 (0.053)
Large graph, $n = 50$	0.646 (0.023)	0.518 (0.008)	0.690 (0.017)
Large graph, $n = 1000$	0.738 (0.039)	0.616 (0.044)	0.832 (0.024)

Table 3: Mean AUC for Contaminated data

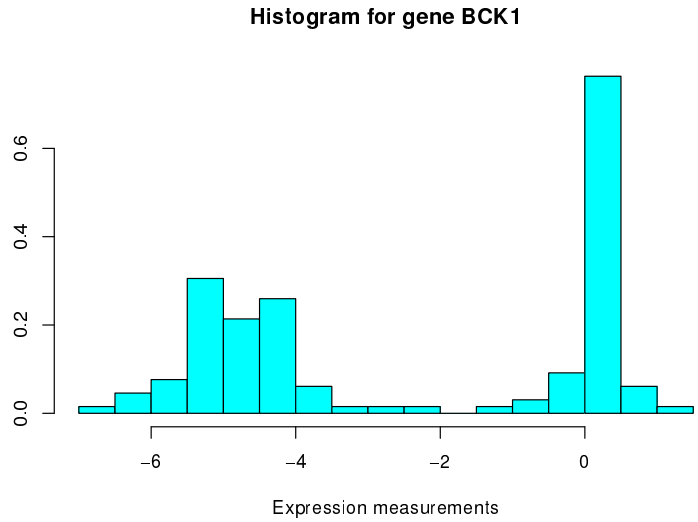


Figure 1: A histogram suggesting a bimodal distribution for the expression values of gene BCK1.

obvious departures from multivariate normality. This said, a Gaussian copula remains of course a strong assumption about the joint distribution.

We ran the PC algorithm using Pearson correlations, Spearman correlations as well as Kendall's τ. We considered a grid of values for α from 10^{-8} to 0.5 and selected α by optimizing the Bayesian information criterion (BIC) of Schwarz (1978). (Extensions in the spirit of Chen and Chen, 2008 and Foygel and Drton, 2010 could be attractive for this tuning problem but have yet to be adapted and studied for directed graphs.) The computations were done using routines from the aforementioned R package `pcalg` as well as the package `ggm` (Sadeghi and Marchetti, 2012). The former package offers, in particular, routines to create DAGs from the PC output and the latter package contains a routine to fit a DAG model by maximum likelihood.

For the case of Pearson correlations, tuning with BIC gave $\alpha = 0.5$ and a graph with 178 edges. Spearman correlations behaved similarly. No true optimum arose during the BIC tuning, which again suggested $\alpha = 0.5$ and led to a graph with 171 edges. For Kendall's τ on the other hand, the BIC was minimal for $\alpha = 0.1$ and only values in the range $[0.05, 0.1]$ gave comparable BIC values. The graph inferred for $\alpha = 0.1$ has 74 edges. We display its largest connected component in Figure 2.

Figure 2 was produced using output from TETRAD IV and features directed, undirected and bidirected edges. While the former two arise in CPDAGs, the latter type of edge indicates inconsistencies that the PC algorithm encountered. Briefly put, a bidirected edge arises when this edge appears in the skeleton inferred in the first stage of the PC algorithm but the edge orientation rules in the second stage of the algorithm yield arrowheads at both tails of the edge.

As mentioned in Sun and Li (2012), some prior biological knowledge about the pathway is available but not in a form that can be translated into a statistical model as considered here. Nevertheless, in this example, the use of Kendall's τ seems preferable to that of Pearson and also Spearman correlations. Both the sparsity of the inferred graph as well as the more favorable behavior in the likelihood computations underlying the BIC search speak for Kendall's τ.

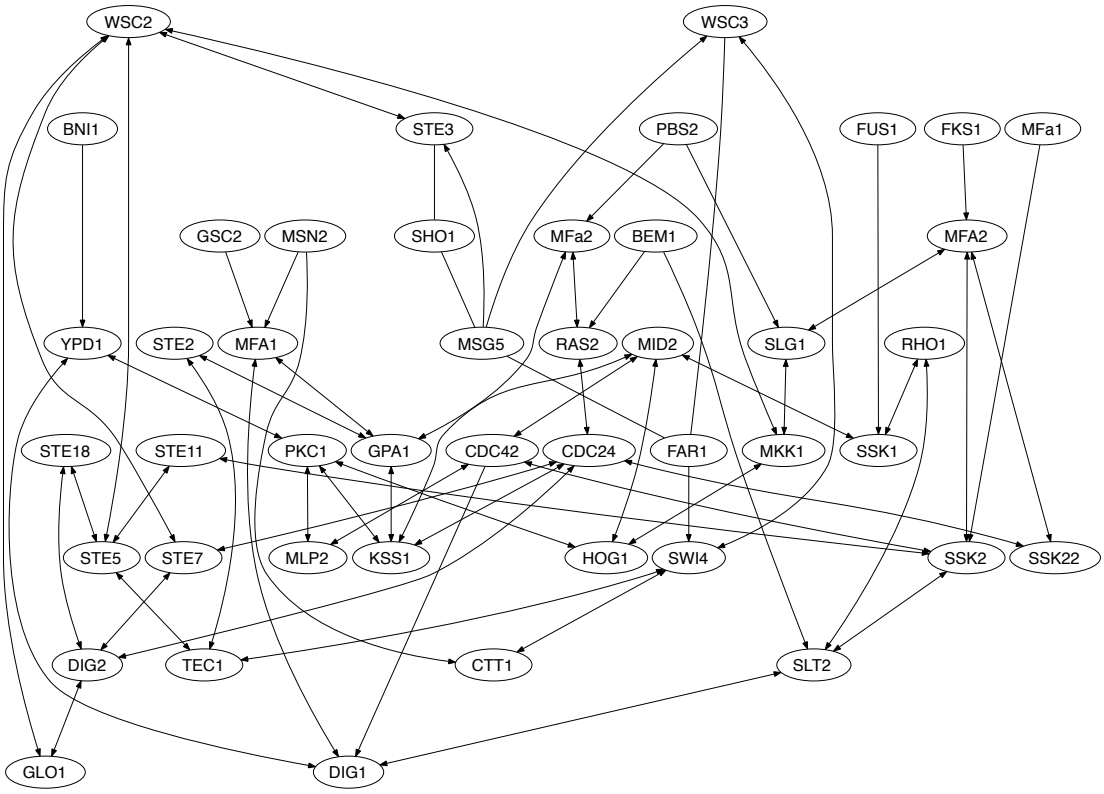


Figure 2: Largest connected component in the output of the Kendall RPC algorithm applied to expression data for genes in the MAPK pathway in yeast.

6. Conclusion

The PC algorithm of Spirtes et al. (2000) addresses the problem of model selection in graphical modelling with directed graphs via a clever scheme of testing conditional independences. For multivariate normal observations, the algorithm is known to have high-dimensional consistency properties when conditional independence is tested using sample partial correlations (Kalisch and Bühlmann, 2007). We showed that the PC algorithm retains these consistency properties when observations follow a Gaussian copula model and rank-based measures of correlation are used to assess conditional independence. The assumptions needed in our analysis are no stronger than those in prior Gaussian work when the considered sequence of DAGs has bounded degree. When the degree grows our assumptions are slightly more restrictive as our proof requires control of the conditioning of principal submatrices of correlation matrices that are inverted to estimate partial correlations in the rank-based PC (RPC) algorithm.

In our simulations, the use of the RPC algorithm led to negligible differences in statistical efficiency when data were indeed normal. For nonnormal data, RPC clearly outperformed the other considered versions of the algorithm. Since rank correlations take only marginally longer to com-

pute than sample correlations, the simulations suggest that there are hardly any downsides associated with making RPC the standard version of the PC algorithm for continuous data.

Consistency results assume the data-generating distribution to be faithful to an underlying DAG. In fact, our results make the stronger assumption that non-zero partial correlations are sufficiently far from zero. As shown in Uhler et al. (2013), this can be a restrictive assumption, which provides an explanation for why consistency does not 'kick-in' quicker in simulation studies such as the one in Kalisch and Bühlmann (2007) and also ours.

Our analysis of the PC algorithm made use of two main arguments. First, for graphs with suitably bounded degree the population version of the PC algorithm only needs to check conditional independences with small conditioning sets. Second, the low-order partial correlations whose vanishing corresponds to these conditional independence can be estimated accurately. Lemma 4, which provides the error propagation from marginal to partial correlations, could similarly be used to analyze other algorithms that test the vanishing of low-order partial correlations. One example is the FCI algorithm that infers a more complex graphical object to deal with situations in which some relevant variables remain unobserved (Spirtes et al., 2000; Colombo et al., 2012).

Recent work shows that Kendall's τ can be used to obtain accurate estimates of the dispersion parameters in a more general setting of elliptical (rather than nonparanormal) distributions. Our analysis would again carry over to this case as an analogue to (5) is available in this setting. However, in the elliptical family zeros in the dispersion matrix do not correspond to independences and would have to be interpreted in terms of a latent normal random vector (Liu et al., 2012b).

Acknowledgments

We would like to thank three anonymous reviewers for carefully reading a draft manuscript and offering insightful suggestions which greatly improved our paper. Mathias Drton was supported by the NSF under Grant No. DMS-0746265 and by an Alfred P. Sloan Fellowship.

Appendix A. Sample Size Adjustment

We now show that the consistency result in Corollary 9 still holds when using the conditional independence tests from (15). In these tests, the sample size is adjusted from n to $n-|S|-3$.

Proof The test in (15) accepts a conditional independence hypothesis if and only if

$$|\hat{\rho}_{uv|S}| \;\leq\; \gamma(n,|S|,z), \tag{17}$$

where

$$\gamma(n,|S|,z) = \frac{\exp\left(z/\sqrt{n-|S|-3}\right)-1}{\exp\left(z/\sqrt{n-|S|-3}\right)+1}$$

and $z = z(\alpha) = 2\Phi^{-1}(1-\alpha/2)$. We need to find a sequence (α_n) of values for α such that consistency holds under the scaling assumptions made in Corollary 9. We will do this by specifying a sequence (z_n) for values for the (doubled) quantiles z.

We claim that the RPC algorithm using the tests from (17) is consistent when choosing the quantile sequence

$$z_n = \sqrt{n-3}\cdot\log\left(\frac{1+c_n/3}{1-c_n/3}\right), \tag{18}$$

where we use the abbreviation

$$c_n := c_{\min}(\Sigma_n, q_n).$$

We will show that as the sample size n tends to infinity, with probability tending to one, $|\hat{\rho}_{uv|S} - \rho_{uv|S}| < c_n/3$ for every $u, v \in V$ and $|S| \leq q_n$. Furthermore, we will show that for the above choice of z_n and all sufficiently large n, we have $c_n/3 \leq \gamma(n, |S|, z_n) \leq 2c_n/3$ for each relevant set S with $0 \leq |S| \leq q_n$. These facts imply that, with asymptotic probability one, every conditional independence test is correct, and the RPC algorithm succeeds.

First, we slightly adapt the proof of Theorem 8. Choosing the uniform error threshold for the correlation estimates as

$$\varepsilon = \frac{c\lambda^2}{(6+c)q + \lambda cq} > 0$$

in place of (13) yields that, with probability at least

$$1 - \frac{A}{2} p^2 \exp\left(-\frac{B\lambda^4 n c^2}{64 q^2}\right), \tag{19}$$

we have that $|\hat{\rho}_{uv|S} - \rho_{uv|S}| < c/3$ for every $u, v \in V$ and $|S| \leq q$. When substituting p_n, q_n, c_n and $\lambda_{\min}(\Sigma_n, q_n)$ for p, q, c and λ, respectively, the scaling assumptions in Corollary 9 imply that the probability bound in (19) tends to one as $n \to \infty$, and we obtain the first part of our claim.

For the second part of our claim, note that our choice of z_n in (18) gives $\gamma(n, 0, z_n) = c_n/3$. Since $\gamma(n, |S|, z)$ is monotonically increasing in $|S|$, we need only show that for sufficiently large n,

$$\gamma(n, q_n, z_n) - \gamma(n, 0, z_n) \leq c_n/3.$$

For $x \geq 0$, the function

$$f(x) = \frac{\exp(x) - 1}{\exp(x) + 1}$$

is concave and, thus, for any $q_n \geq 0$,

$$\begin{aligned} \gamma(n, q_n, z_n) - \gamma(n, 0, z_n) &= f\left(\frac{z}{\sqrt{n - q_n - 3}}\right) - f\left(\frac{z}{\sqrt{n-3}}\right) \\ &\leq f'\left(\frac{z}{\sqrt{n-3}}\right)\left(\frac{z}{\sqrt{n-q_n-3}} - \frac{z}{\sqrt{n-3}}\right). \end{aligned} \tag{20}$$

The derivative of f is

$$f'(x) = \frac{2\exp(x)}{(\exp(x)+1)^2}.$$

Evaluating the right hand side of (20), we obtain that

$$\begin{aligned} \gamma(n, q_n, z_n) - \gamma(n, 0, z_n) &\leq \frac{1}{2}\left(1 - \frac{c_n^2}{9}\right) \log\left(\frac{1 + c_n/3}{1 - c_n/3}\right)\left(\frac{\sqrt{n-3}}{\sqrt{n-q_n-3}} - 1\right) \\ &\leq \frac{1}{2} \log\left(\frac{1 + c_n/3}{1 - c_n/3}\right)\left(\frac{\sqrt{n-3}}{\sqrt{n-q_n-3}} - 1\right). \end{aligned} \tag{21}$$

Being derived from absolute values of partial correlations, the sequence c_n is in $[0,1]$. Now, $\log[(1+x)/(1-x)]$ is a convex function of $x \geq 0$ that is zero at $x = 0$ and equal to $\log(2)$ for $x = 1/3$. Therefore,

$$\frac{1}{2}\log\left(\frac{1+c_n/3}{1-c_n/3}\right) \leq \frac{1}{2}\log(2)\cdot c_n, \qquad c_n \in [0,1].$$

This shows that the bound in (21) is $o(c_n)$ because, by assumption, $q_n = o(\sqrt{n})$. In particular, the bound in (21) is less than $c_n/3$ for sufficiently large n, proving the claimed consistency result. ■

Appendix B. Background on Graphical Models

Let $G = (V,E)$ be an acyclic digraph with finite vertex set. We write $v \to w \in E$ to indicate that (v,w) is an edge in E. As mentioned in the introduction, the conditional independences associated with the graph G may be determined using d-separation; compare, for example, page 48 in Lauritzen (1996). We briefly review the concept.

Since a DAG contains at most one edge between any two nodes, we may define a path from a node u to a node v to be a sequence of distinct nodes $(v_0, v_1, \ldots, v_n)$ such that $v_0 = u$, $v_n = v$ and for all $1 \leq k \leq n$, either $v_{k-1} \to v_k \in E$ or $v_{k-1} \leftarrow v_k \in E$. Two distinct nodes u and v are then said to be *d-separated* by a set $S \subset V \setminus \{v,u\}$ if every path from u to v contains three consecutive nodes (v_{k-1}, v_k, v_{k+1}) for which one of the following is true:

(i) The three nodes form a chain $v_{k-1} \to v_k \to v_{k+1}$, a chain $v_{k-1} \leftarrow v_k \leftarrow v_{k+1}$, or a fork $v_{k-1} \leftarrow v_k \to v_{k+1}$, and the middle node v_k is in S.

(ii) The three nodes form a collider $v_{k-1} \to v_k \leftarrow v_{k+1}$, and neither v_k nor any of its descendants is in S.

Suppose A, B, S are pairwise disjoint subsets of V. Then S d-separates A and B if S d-separates any pair of nodes a and b with $a \in A$ and $b \in B$.

Two DAGs $G = (V,E)$ and $H = (V,F)$ with the same vertex set V are *Markov equivalent* if they may possess the same d-separation relations, that is, two sets A and B are d-separated given a third set C in the graph G if and only if the same holds in H. To give an example, the graphs $u \to v \to w$ and $u \leftarrow v \leftarrow w$ are Markov equivalent, but $u \to v \to w$ and $u \to v \leftarrow w$ are not. As first shown in Verma and Pearl (1991), two DAGs G and H are Markov equivalent if and only if they have the same skeleton and the same unshielded colliders. The *skeleton* of a digraph G is the undirected graph obtained by converting each directed edge into an undirected edge. An *unshielded collider* is a triple of nodes (u,v,w) that induces the subgraph $u \to v \leftarrow w$, that is, there is no edge between u and w.

Let $[G]$ be the Markov equivalence class of an acyclic digraph $G = (V,E)$. Write $E(H)$ for the edge set of a DAG H, and define the edge set

$$[E] = \bigcup_{H \in [G]} E(H).$$

That is, $(v,w) \in [E]$ if there exists a DAG $H \in [G]$ with the edge $v \to w$ in its edge set. We interpret the presence of both (v,w) and (w,v) in $[E]$ as an undirected edge between v and w. The graph

$C(G) = (V, [E])$ is known as the *completed partially directed acyclic graph* (CPDAG) for G or also as the *essential graph*. Two DAGs G and H satisfy $C(G) = C(H)$ if and only if $[G] = [H]$, making the CPDAG a useful graphical representation of a Markov equivalence class; see Andersson et al. (1997) and Chickering (2002).

References

Theodore. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley Series in Probability and Statistics. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, third edition, 2003.

Steen A. Andersson, David Madigan, and Michael D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *Ann. Statist.*, 25(2):505–541, 1997.

Rachel B. Brem and Leonid Kruglyak. The landscape of genetic complexity across 5,700 gene expression traits in yeast. *Proceedings of National Academy of Sciences*, 102:1572–1577, 2005.

Jiahua Chen and Zehua Chen. Extended Bayesian information criterion for model selection with large model sp ace. *Biometrika*, 95:759–771, 2008.

David Maxwell Chickering. Learning equivalence classes of Bayesian-network structures. *J. Mach. Learn. Res.*, 2(3):445–498, 2002.

David Christensen. Fast algorithms for the calculation of Kendall's τ. *Comput. Statist.*, 20(1):51–62, 2005.

Diego Colombo, Marloes H. Maathuis, Markus Kalisch, and Thomas S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *Ann. Statist.*, 40(1): 294–321, 2012.

Mathias Drton, Bernd Sturmfels, and Seth Sullivant. *Lectures on Algebraic Statistics*, volume 39 of *Oberwolfach Seminars*. Birkhäuser Verlag, Basel, 2009.

Rina Foygel and Mathias Drton. Extended Bayesian information criteria for Gaussian graphical models. *Adv. Neural Inf. Process. Syst.*, 23:2020–2028, 2010.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1990. Corrected reprint of the 1985 original.

Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *J. Mach. Learn. Res.*, 8:613–636, May 2007.

Markus Kalisch and Peter Bühlmann. Robustification of the PC-algorithm for directed acyclic graphs. *J. Comput. Graph. Statist.*, 17(4):773–789, 2008.

Markus Kalisch, Martin Mächler, Diego Colombo, Marloes H. Maathuis, and Peter Bühlmann. Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26, 5 2012.

Steffen L. Lauritzen. *Graphical Models*, volume 17 of *Oxford Statistical Science Series*. The Clarendon Press Oxford University Press, New York, 1996.

Thuc Duy Le, Lin Liu, Anna Tsykin, Gregory J. Goodall, Bing Liu, Bing-Yu Sun, and Jiuyong Li. Inferring microRNA-mRNA causal regulatory relationships from expression data. *Bioinformatics*, 29(6):765–771, 2013.

Han Liu, John Lafferty, and Larry Wasserman. The nonparanormal: semiparametric estimation of high dimensional undirected graphs. *J. Mach. Learn. Res.*, 10:2295–2328, 2009.

Han Liu, Fang Han, Ming Yuan, John Lafferty, and Larry Wasserman. High-dimensional semiparametric Gaussian copula graphical models. *Ann. Statist.*, 40(4):2293–2326, 2012a.

Han Liu, Fang Han, and Cun-hui Zhang. Transelliptical graphical models. *Adv. Neural Inf. Process. Syst.*, 25:800–808, 2012b.

Marloes H. Maathuis, Diego Colombo, Markus Kalisch, and Peter Bühlmann. Predicting causal effects in large-scale systems from observational data. *Nature Methods*, 7(4):247–248, 2010.

Judea Pearl. *Causality*. Cambridge University Press, Cambridge, second edition, 2009. Models, reasoning, and inference.

Kayvan Sadeghi and Giovanni M. Marchetti. Graphical Markov models with mixed graphs in R. *The R Journal*, 4(2):65–73, December 2012.

Markus Schmidberger, Sabine Lennert, and Ulrich Mansmann. Conceptual aspects of large meta-analyses with publicly available microarray data: A case study in oncology. *Bioinformatics and Biology Insights*, 5:13–39, 2011.

Gideon Schwarz. Estimating the dimension of a model. *Ann. Statist.*, 6(2):461–464, 1978.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, second edition, 2000. With additional material by David Heckerman, Christopher Meek, Gregory F. Cooper and Thomas Richardson, A Bradford Book.

Hokeun Sun and Hongzhe Li. Robust Gaussian graphical modeling via l_1 penalization. *Biometrics*, 68:1197–1206, 2012.

Caroline Uhler, Garvesh Raskutti, Bin Yu, and Peter Bühlmann. Geometry of faithfulness assumption in causal inference. *Ann. Statist.*, 41(2):436–463, 2013.

Ricardo A. Verdugo, Tanja Zeller, Maxime Rotival, Philipp S. Wild, Thomas Münzel, Karl J. Lackner, Henri Weidmann, Ewa Ninio, David-Alexandre Trégouët, François Cambien, Stefan Blankenberg, and Laurence Tiret. Graphical modeling of gene expression in monocytes suggests molecular mechanisms explaining increased atherosclerosis in smokers. *PLoS ONE*, 8(1): e50888, 2013.

Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. Technical Report R-150, UCLA, 1991.

Journal of Machine Learning Research 14 (2013) 3385-3418 Submitted 2/12; Revised 8/13; Published 11/13

Sparse Matrix Inversion with Scaled Lasso

Tingni Sun TINGNI@WHARTON.UPENN.EDU
Statistics Department
The Wharton School
University of Pennsylvania
Philadelphia, Pennsylvania, 19104, USA

Cun-Hui Zhang CZHANG@STAT.RUTGERS.EDU
Department of Statistics and Biostatistics
Rutgers University
Piscataway, New Jersey 08854, USA

Editor: Bin Yu

Abstract

We propose a new method of learning a sparse nonnegative-definite target matrix. Our primary example of the target matrix is the inverse of a population covariance or correlation matrix. The algorithm first estimates each column of the target matrix by the scaled Lasso and then adjusts the matrix estimator to be symmetric. The penalty level of the scaled Lasso for each column is completely determined by data via convex minimization, without using cross-validation.

We prove that this scaled Lasso method guarantees the fastest proven rate of convergence in the spectrum norm under conditions of weaker form than those in the existing analyses of other ℓ_1 regularized algorithms, and has faster guaranteed rate of convergence when the ratio of the ℓ_1 and spectrum norms of the target inverse matrix diverges to infinity. A simulation study demonstrates the computational feasibility and superb performance of the proposed method.

Our analysis also provides new performance bounds for the Lasso and scaled Lasso to guarantee higher concentration of the error at a smaller threshold level than previous analyses, and to allow the use of the union bound in column-by-column applications of the scaled Lasso without an adjustment of the penalty level. In addition, the least squares estimation after the scaled Lasso selection is considered and proven to guarantee performance bounds similar to that of the scaled Lasso.

Keywords: precision matrix, concentration matrix, inverse matrix, graphical model, scaled Lasso, linear regression, spectrum norm

1. Introduction

We consider the estimation of the matrix inversion Θ^* satisfying $\overline{\Sigma}\Theta^* \approx I$ for a given data matrix $\overline{\Sigma}$. When $\overline{\Sigma}$ is a sample covariance matrix, our problem is the estimation of the inverse of the corresponding population covariance matrix. The inverse covariance matrix is also called precision matrix or concentration matrix. With the dramatic advances in technology, the number of variables p, or the size of the matrix Θ^*, is often of greater order than the sample size n in statistical and engineering applications. In such cases, the sample covariance matrix is always singular and a certain type of sparsity condition is typically imposed for proper estimation of the precision matrix and for theoretical investigation of the problem. In a simple version of our theory, this condition is

expressed as the ℓ_0 sparsity, or equivalently the maximum degree, of the target inverse matrix Θ^*. A weaker condition of capped ℓ_1 sparsity is also studied to allow many small signals.

Several approaches have been proposed to the estimation of sparse inverse matrices in high-dimensional setting. The ℓ_1 penalization is one of the most popular methods. Lasso-type methods, or convex minimization algorithms with the ℓ_1 penalty on all entries of Θ^*, have been developed in Banerjee et al. (2008) and Friedman et al. (2008), and in Yuan and Lin (2007) with ℓ_1 penalization on the off-diagonal matrix only. This is refereed to as the graphical Lasso (GLasso) due to the connection of the precision matrix to Gaussian Markov graphical models. In this GLasso framework, Ravikumar et al. (2008) provides sufficient conditions for model selection consistency, while Rothman et al. (2008) provides the convergence rate $\{((p+s)/n)\log p\}^{1/2}$ in the Frobenius norm and $\{(s/n)\log p\}^{1/2}$ in the spectrum norm, where s is the number of nonzero off-diagonal entries in the precision matrix. Concave penalty has been studied to reduce the bias of the GLasso (Lam and Fan, 2009). Similar convergence rates have been studied under the Frobenius norm in a unified framework for penalized estimation in Negahban et al. (2012). Since the spectrum norm can be controlled via the Frobenius norm, this provides a sufficient condition $(s/n)\log p \to 0$ for the convergence to the unknown precision matrix under the spectrum norm. However, in the case of $p \geq n$, this condition does not hold for banded precision matrices, where s is of the order of the product of p and the width of the band.

A potentially faster rate $d\sqrt{(\log p)/n}$ can be achieved by ℓ_1 regularized estimation of individual columns of the precision matrix, where d, the matrix degree, is the largest number of nonzero entries in a column. This was done in Yuan (2010) by applying the Dantzig selector to the regression of each variable against others, followed by a symmetrization step via linear programming. When the ℓ_1 operator norm of the precision matrix is bounded, this method achieves the convergence rate $d\sqrt{(\log p)/n}$ in ℓ_q matrix operator norms. The CLIME estimator (Cai et al., 2011), which uses the Dantzig selector directly to estimate each column of the precision matrix, also achieves the $d\sqrt{(\log p)/n}$ rate under the boundedness assumption of the ℓ_1 operator norm. In Yang and Kolaczyk (2010), the Lasso is applied to estimate the columns of the target matrix under the assumption of equal diagonal, and the estimation error is studied in the Frobenius norm for $p = n^{\nu}$. This column-by-column idea reduces a graphical model to p regression models. It was first introduced by Meinshausen and Bühlmann (2006) for identifying nonzero variables in a graphical model, called neighborhood selection. In addition, Rocha et al. (2008) proposed a pseudo-likelihood method by merging all p linear regressions into a single least squares problem.

In this paper, we propose to apply the scaled Lasso (Sun and Zhang, 2012) column-by-column to estimate a precision matrix in the high dimensional setting. Based on the connection of precision matrix estimation to linear regression, we construct a column estimator with the scaled Lasso, a joint estimator for the regression coefficients and noise level. Since we only need a sample covariance matrix as input, this estimator could be extended to generate an approximate inverse of a nonnegative-definite data matrix in a more general setting. This scaled Lasso algorithm provides a fully specified map from the space of nonnegative-definite matrices to the space of symmetric matrices. For each column, the penalty level of the scaled Lasso is determined by data via convex minimization, without using cross-validation.

We study theoretical properties of the proposed estimator for a precision matrix under a normality assumption. More precisely, we assume that the data matrix is the sample covariance matrix $\overline{\Sigma} = X^T X/n$, where the rows of X are iid $N(0,\Sigma^*)$ vectors. Let $R^* = (\mathrm{diag}\Sigma^*)^{-1/2}\Sigma^*(\mathrm{diag}\Sigma^*)^{-1/2}$ be the population correlation matrix. Our target is to estimate the inverse matrices $\Theta^* = (\Sigma^*)^{-1}$ and

$\Omega^* = (R^*)^{-1}$. Define

$$d = \max_{1 \le j \le p} \#\{k : \Theta^*_{jk} \neq 0\}. \tag{1}$$

A simple version of our main theoretical result can be stated as follows.

Theorem 1 *Let $\widehat{\Theta}$ and $\widehat{\Omega}$ be the scaled Lasso estimators defined in (4), (7) and (9) below with penalty level $\lambda_0 = A\sqrt{4(\log p)/n}$, $A > 1$, based on n iid observations from $N(0, \Sigma^*)$. Suppose the spectrum norm of $\Omega^* = (\mathrm{diag}\Sigma^*)^{1/2}\Theta^*(\mathrm{diag}\Sigma^*)^{1/2}$ is bounded and that $d^2(\log p)/n \to 0$. Then,*

$$\|\widehat{\Omega} - \Omega^*\|_2 = O_P(1)d\sqrt{(\log p)/n} = o(1),$$

where $\|\cdot\|_2$ is the spectrum norm (the ℓ_2 matrix operator norm). If in addition the diagonal elements of Θ^* is uniformly bounded, then

$$\|\widehat{\Theta} - \Theta^*\|_2 = O_P(1)d\sqrt{(\log p)/n} = o(1).$$

Theorem 1 provides a simple boundedness condition on the spectrum norm of Ω^* for the convergence of $\widehat{\Omega}$ in spectrum norm with sample size $n \gg d^2 \log p$. The additional condition on the diagonal of Θ^* is natural due to scale change. The boundedness condition on the spectrum norm of $(\mathrm{diag}\Sigma^*)^{1/2}\Theta^*(\mathrm{diag}\Sigma^*)^{1/2}$ and the diagonal of Θ^* is weaker than the boundedness of the ℓ_1 operator norm assumed in Yuan (2010) and Cai et al. (2011) since the boundedness of $\mathrm{diag}\Sigma^*$ is also needed there. When the ratio of the ℓ_1 operator norm and spectrum norm of the precision matrix diverges to infinity, the proposed estimator has a faster proven convergence rate. This sharper result is a direct consequence of the faster convergence rate of the scaled Lasso estimator of the noise level in linear regression. To the best of our knowledge, it is unclear if the ℓ_1 regularization method of Yuan (2010) and Cai et al. (2011) also achieve the convergence rate under the weaker spectrum norm condition.

An important advantage of the scaled Lasso is that the penalty level is automatically set to achieve the optimal convergence rate in the regression model for the estimation of each column of the inverse matrix. This raises the possibility for the scaled Lasso to outperform methods using a single unscaled penalty level for the estimation of all columns such as the GLasso and CLIME. We provide an example in Section 7 to demonstrate the feasibility of such a scenario.

Another contribution of this paper is to study the scaled Lasso at a smaller penalty level than those based on ℓ_∞ bounds of the noise. The ℓ_∞-based analysis requires a penalty level λ_0 satisfying $P\{N(0,1/n) > \lambda_0/A\} = \varepsilon/p$ for a small ε and $A > 1$. For $A \approx 1$ and $\varepsilon = p^{o(1)}$, this penalty level is comparable to the universal penalty level $\sqrt{(2/n)\log p}$. However, $\varepsilon = o(1/p)$, or equivalently $\lambda_0 \approx \sqrt{(4/n)\log p}$, is required if the union bound is used to simultaneously control the error of p applications of the scaled Lasso in the estimation of individual columns of a precision matrix. This may create a significant gap between theory and implementation. We close this gap by providing a theory based on a sparse ℓ_2 measure of the noise, corresponding to a penalty level satisfying $P\{N(0,1/n) > \lambda_0/A\} = k/p$ with $A > 1$ and a potentially large k. This penalty level provides a faster convergence rate than the universal penalty level in linear regression when $\log(p/k) \approx \log(p/\|\beta\|_0) \ll \log p$. Moreover, the new analysis provides a higher concentration of the error so that the same penalty level $\lambda_0 \approx \sqrt{(2/n)\log(p/k)}$ can be used to simultaneously control the estimation error in p applications of the scaled Lasso for the estimation of a precision matrix.

The rest of the paper is organized as follows. In Section 2, we present the scaled Lasso method for the estimation of the inversion of a nonnegative definite matrix. In Section 3, we study the estimation error of the proposed method. In Section 4, we provide a theory for the Lasso and its scaled version with higher proven concentration at a smaller, practical penalty level. In Section 5, we study the least square estimation after the scaled Lasso selection. Simulation studies are presented in Section 6. In Section 7, we discuss the benefits of using the scaled penalty levels for the estimation of different columns of the precision matrix, compared with an optimal fixed penalty level for all columns. An appendix provides all the proofs.

We use the following notation throughout the paper. For real x, $x_+ = \max(x,0)$. For a vector $v = (v_1,\ldots,v_p)$, $\|v\|_q = (\sum_j |v_j|^q)^{1/q}$ is the ℓ_q norm with the special $\|v\| = \|v\|_2$ and the usual extensions $\|v\|_\infty = \max_j |v_j|$ and $\|v\|_0 = \#\{j : v_j \neq 0\}$. For matrices M, $M_{i,*}$ is the i-th row and $M_{*,j}$ the j-th column, $M_{A,B}$ represents the submatrix of M with rows in A and columns in B, $\|M\|_q = \sup_{\|v\|_q=1} \|Mv\|_q$ is the ℓ_q matrix operator norm. In particular, $\|\cdot\|_2$ is the spectrum norm for symmetric matrices. Moreover, we may denote the set $\{j\}$ by j and denote the set $\{1,\ldots,p\}\setminus\{j\}$ by $-j$ in the subscript.

2. Matrix Inversion via Scaled Lasso

Let $\overline{\Sigma}$ be a nonnegative-definite data matrix and Θ^* be a positive-definite target matrix with $\overline{\Sigma}\Theta^* \approx I$. In this section, we describe the relationship between positive-definite matrix inversion and linear regression and propose an estimator for Θ^* via scaled Lasso, a joint convex minimization for the estimation of regression coefficients and noise level.

We use the scaled Lasso to estimate Θ^* column by column. Define $\sigma_j > 0$ and $\beta \in \mathbb{R}^{p\times p}$ by

$$\sigma_j^2 = (\Theta^*_{jj})^{-1}, \quad \beta_{*,j} = -\Theta^*_{*,j}\sigma_j^2 = -\Theta^*_{*,j}(\Theta^*_{jj})^{-1}.$$

In the matrix form, we have the following relationship

$$\mathrm{diag}\Theta^* = \mathrm{diag}(\sigma_j^{-2}, j = 1,\ldots,p), \quad \Theta^* = -\beta(\mathrm{diag}\Theta^*). \tag{2}$$

Let $\Sigma^* = (\Theta^*)^{-1}$. Since $(\partial/\partial b_{-j}) b^T\Sigma^* b = 2\Sigma^*_{-j,*} b = 0$ at $b = \beta_{*,j}$, one may estimate the j-th column of β by minimizing the ℓ_1 penalized quadratic loss. In order to penalize the unknown coefficients in the same scale, we adjust the ℓ_1 penalty with diagonal standardization, leading to the following penalized quadratic loss:

$$b^T\overline{\Sigma}b/2 + \lambda\sum_{k=1}^p \overline{\Sigma}_{kk}^{1/2}|b_k|. \tag{3}$$

For $\overline{\Sigma} = X^TX/n$ and $b_j = -1$, $b^T\overline{\Sigma}b = \|x_j - \sum_{k\neq j} b_k x_k\|_2^2/n$, so that (3) is the penalized loss for the Lasso in linear regression of x_j against $\{x_k, k \neq j\}$. This is similar to the procedures in Yuan (2010) and Cai et al. (2011) that use the Dantzig selector to estimate $\Theta^*_{*,j}$ column-by-column. However, one still needs to choose a penalty level λ and to estimate σ_j in order to recover Θ^* via (2). A solution to resolve these two issues is the scaled Lasso (Sun and Zhang, 2012):

$$\{\widehat{\beta}_{*,j}, \widehat{\sigma}_j\} = \arg\min_{b,\sigma}\Big\{\frac{b^T\overline{\Sigma}b}{2\sigma} + \frac{\sigma}{2} + \lambda_0\sum_{k=1}^p \overline{\Sigma}_{kk}^{1/2}|b_k| : b_j = -1\Big\} \tag{4}$$

with $\lambda_0 \approx \sqrt{(2/n)\log p}$. The scaled Lasso (4) is a solution of joint convex minimization in $\{b, \sigma\}$ (Huber and Ronchetti, 2009; Antoniadis, 2010). Since $\beta^T \Sigma^* \beta = (\mathrm{diag}\Theta^*)^{-1}\Theta^*(\mathrm{diag}\Theta^*)^{-1}$,

$$\mathrm{diag}\big(\beta^T \Sigma^* \beta\big) = (\mathrm{diag}\Theta^*)^{-1} = \mathrm{diag}(\sigma_j^2, j = 1, \ldots, p).$$

Thus, (4) is expected to yield consistent estimates of $\sigma_j = (\Theta^*_{jj})^{-1/2}$.

An iterative algorithm has been provided in Sun and Zhang (2012) to compute the scaled Lasso estimator (4). We rewrite the algorithm in the form of matrices. For each $j \in \{1, \ldots, p\}$, the Lasso path is given by the estimates $\widehat{\beta}_{-j,j}(\lambda)$ satisfying the following Karush-Kuhn-Tucker conditions: for all $k \neq j$,

$$\begin{cases} \overline{\Sigma}_{kk}^{-1/2}\overline{\Sigma}_{k,*}\widehat{\beta}_{*,j}(\lambda) = -\lambda \mathrm{sgn}(\widehat{\beta}_{k,j}(\lambda)), & \widehat{\beta}_{k,j} \neq 0, \\ \overline{\Sigma}_{kk}^{-1/2}\overline{\Sigma}_{k,*}\widehat{\beta}_{*,j}(\lambda) \in \lambda[-1,1], & \widehat{\beta}_{k,j} = 0, \end{cases} \tag{5}$$

where $\widehat{\beta}_{jj}(\lambda) = -1$. Based on the Lasso path $\widehat{\beta}_{*,j}(\lambda)$, the scaled Lasso estimator $\{\widehat{\beta}_{*,j}, \widehat{\sigma}_j\}$ is computed iteratively by

$$\widehat{\sigma}_j^2 \leftarrow \widehat{\beta}_{*,j}^T \overline{\Sigma}\widehat{\beta}_{*,j}, \quad \lambda \leftarrow \widehat{\sigma}_j \lambda_0, \quad \widehat{\beta}_{*,j} \leftarrow \widehat{\beta}_{*,j}(\lambda). \tag{6}$$

Here the penalty level of the Lasso is determined by the data without using cross-validation. We then simply take advantage of the relationship (2) and compute the coefficients and noise levels by the scaled Lasso for each column

$$\mathrm{diag}\widetilde{\Theta} = \mathrm{diag}(\widehat{\sigma}_j^{-2}, j = 1, \ldots, p), \quad \widetilde{\Theta} = -\widehat{\beta}(\mathrm{diag}\widetilde{\Theta}). \tag{7}$$

Now we have constructed an estimator for Θ^*. In our primary example of taking $\overline{\Sigma}$ as a sample covariance matrix, the target Θ^* is the inverse covariance matrix. One may also be interested in estimating the inverse correlation matrix

$$\Omega^* = (R^*)^{-1} = \big\{D^{-1/2}\Sigma^* D^{-1/2}\big\}^{-1} = D^{1/2}(\Sigma^*)^{-1}D^{1/2}, \tag{8}$$

where $D = \mathrm{diag}(\Sigma^*)$ and $R^* = D^{-1/2}\Sigma^* D^{-1/2}$ is the population correlation matrix. The diagonal matrix D can be approximated by the diagonal of $\overline{\Sigma}$. Thus, the inverse correlation matrix is estimated by

$$\widetilde{\Omega} \;\; = \;\; \widehat{D}^{1/2}\widetilde{\Theta}\widehat{D}^{1/2} \text{ with } \widehat{D} = \mathrm{diag}(\overline{\Sigma}_{jj}, j = 1, \ldots, p).$$

The estimator $\widetilde{\Omega}$ here is a result of normalizing the precision matrix estimator by the population variances. Alternatively, we may estimate the inverse correlation matrix by using the population correlation matrix

$$\overline{R} = (\mathrm{diag}\overline{\Sigma})^{-1/2}\overline{\Sigma}(\mathrm{diag}\overline{\Sigma})^{-1/2} = \widehat{D}^{-1/2}\overline{\Sigma}\widehat{D}^{-1/2}$$

as data matrix. Let $\{\widehat{\alpha}_{*,j}, \widehat{\tau}_j\}$ be the solution of (4) with $\overline{R}$ in place of $\overline{\Sigma}$. We combine these column estimators as in (7) to have an alternative estimator for Ω^* as follows:

$$\mathrm{diag}\big(\widetilde{\Omega}^{\mathrm{Alt}}\big) = \mathrm{diag}(\widehat{\tau}_j^{-2}, j = 1, \ldots, p), \quad \widetilde{\Omega}^{\mathrm{Alt}} = -\widehat{\alpha}\,\mathrm{diag}\big(\widetilde{\Omega}^{\mathrm{Alt}}\big).$$

Since $\overline{R}_{jj} = 1$ for all j, it follows from (4) that

$$\widehat{\alpha}_{*,j} = \widehat{D}^{1/2}\widehat{\beta}_{*,j}\widehat{D}_{jj}^{-1/2}, \quad \widehat{\tau}_j = \widehat{\sigma}_j\widehat{D}_{jj}^{-1/2}.$$

This implies

$$\widetilde{\Omega}^{\mathrm{Alt}} = -\widehat{D}^{1/2}\widehat{\beta}\mathrm{diag}(\widehat{D}_{jj}^{-1/2}\widehat{\sigma}_j^{-2}\widehat{D}_{jj}, j = 1,\ldots,p) = \widehat{D}^{1/2}\widetilde{\Theta}\widehat{D}^{1/2} = \widetilde{\Omega}.$$

Thus, in this scaled Lasso approach, the estimator based on the normalized data matrix is exactly the same as the one based on the original data matrix followed by a normalization step. The scaled Lasso methodology is scale-free in the noise level, and as a result, the estimator for inverse correlation matrix is also scale free in diagonal normalization.

It is noticed that a good estimator for Θ^* or Ω^* should be a symmetric matrix. However, the estimators $\widetilde{\Theta}$ and $\widetilde{\Omega}$ do not have to be symmetric. We improve them by using a symmetrization step as in Yuan (2010),

$$\widehat{\Theta} = \mathop{\arg\min}_{M:M^T=M} \|M - \widetilde{\Theta}\|_1, \quad \widehat{\Omega} = \mathop{\arg\min}_{M:M^T=M} \|M - \widetilde{\Omega}\|_1, \tag{9}$$

which can be solved by linear programming. It is obvious that $\widehat{\Theta}$ and $\widehat{\Omega}$ are both symmetric, but not guaranteed to be positive-definite. It follows from Theorem 1 that $\widehat{\Theta}$ and $\widehat{\Omega}$ are positive-definite with large probability. Alternatively, semidefinite programming, which is somewhat more expensive computationally, can be used to produce a nonnegative-definite $\widehat{\Theta}$ and $\widehat{\Omega}$ in (9).

According to the definition, the estimators $\widehat{\Theta}$ and $\widehat{\Omega}$ have the same ℓ_1 error rate as $\widetilde{\Theta}$ and $\widetilde{\Omega}$ respectively. A nice property of symmetric matrices is that the spectrum norm is bounded by the ℓ_1 matrix norm. The ℓ_1 matrix norm can be expressed more explicitly as the maximum ℓ_1 norm of the columns, while the ℓ_∞ matrix norm is the maximum ℓ_1 norm of the rows. Hence, for any symmetric matrix, the ℓ_1 matrix norm is equivalent to the ℓ_∞ matrix norm, and the spectrum norm can be bounded by either of them. Since our estimators and target matrices are all symmetric, the error bound based on the spectrum norm could be studied by bounding the ℓ_1 error as typically done in the existing literature. We will study the estimation error of (9) in Section 3.

To sum up, we propose to estimate the matrix inversion by (4), (7) and (9). The iterative algorithm (6) computes (4) based on a Lasso path determined by (5). Then (7) translates the resulting estimators of (6) to column estimators and thus a preliminary matrix estimator is constructed. Finally, the symmetrization step (9) produces a symmetric estimate for our target matrix.

3. Theoretical Properties

From now on, we suppose that the data matrix is the sample covariance matrix $\overline{\Sigma} = X^T X/n$, where the rows of X are iid $N(0,\Sigma^*)$. Let $\Theta^* = (\Sigma^*)^{-1}$ be the precision matrix as the inverse of the population covariance matrix. Let D be the diagonal of Σ^*, $R^* = D^{-1/2}\Sigma^* D^{-1/2}$ the population correlation matrix, $\Omega^* = (R^*)^{-1}$ its inverse as in (8). In this section, we study $\widehat{\Omega}$ and $\widehat{\Theta}$ in (9), respectively for the estimation of Ω^* and Θ^*.

We consider a certain capped ℓ_1 sparsity for individual columns of the inverse matrix as follows. For a certain $\varepsilon_0 > 0$, a threshold level $\lambda_{*,0} > 0$ not depending on j and an index set $S_j \subset \{1,\ldots,p\} \setminus \{j\}$, the capped ℓ_1 sparsity condition measures the complexity of the j-th column of Ω^* by

$$|S_j| + (1-\varepsilon_0)^{-1} \sum_{k\neq j, k\notin S_j} \frac{|\Omega^*_{kj}|}{(\Omega^*_{jj})^{1/2}\lambda_{*,0}} \le s_{*,j}. \tag{10}$$

The condition can be written as

$$\sum_{j\neq k} \min\left\{ \frac{|\Omega^*_{kj}|}{(1-\varepsilon_0)(\Omega^*_{jj})^{1/2}\lambda_{*,0}}, 1 \right\} \leq s_{*,j}$$

if we do not care about the choice of S_j. In the ℓ_0 sparsity case of $S_j = \{k : k \neq j, \Omega^*_{kj} \neq 0\}$, we may set $s_{*,j} = |S_j| + 1$ as the degree for the j-th node in the graph induced by matrix Ω^* (or Θ^*). In this case, $d = \max_j(1 + |S_j|)$ is the maximum degree as in (1).

In addition to the sparsity condition on the inverse matrix, we also require a certain invertibility condition on R^*. Let $S_j \subseteq B_j \subseteq \{1,\ldots,p\} \setminus \{j\}$. A simple version of the required invertibility condition can be written as

$$\inf\left\{ \frac{u^T(R^*_{-j,-j})u}{\|u_{B_j}\|_2^2} : u_{B_j} \neq 0 \right\} \geq c_* \tag{11}$$

with a fixed constant $c_* > 0$. This condition requires a certain partial invertibility of the population correlation matrix. It certainly holds if the smallest eigenvalue of $R^*_{-j,-j}$ is no smaller than c_* for all $j \leq p$, or the spectrum norm of Ω^* is no greater than $1/c_*$ as assumed in Theorem 1. In the proof of Theorems 2 and 3, we only use a weaker version of condition (11) in the form of (35) with $\{\Sigma^*, \overline{\Sigma}\}$ replaced by $\{R^*_{-j,-j}, \overline{R}_{-j,-j}\}$ there.

Theorem 2 *Suppose $\overline{\Sigma}$ is the sample covariance matrix of n iid $N(0,\Sigma^*)$ vectors. Let $\Theta^* = (\Sigma^*)^{-1}$ and Ω^* as in (8) be the inverses of the population covariance and correlation matrices. Let $\widehat{\Theta}$ and $\widehat{\Omega}$ be their scaled Lasso estimators defined in (4), (7) and (9) with a penalty level $\lambda_0 = A\sqrt{4(\log p)/n}$, $A > 1$. Suppose (10) and (11) hold with $\varepsilon_0 = 0$ and $\max_{j\leq p}(1 + s_{*,j})\lambda_0 \leq c_0$ for a certain constant $c_0 > 0$ depending on c_* only. Then, the spectrum norm of the errors are bounded by*

$$\|\widehat{\Theta} - \Theta^*\|_2 \leq \|\widehat{\Theta} - \Theta^*\|_1 \leq C\Big(\max_{j\leq p}(\big\|D_{-j}^{-1}\|_\infty \Theta^*_{jj})^{1/2} s_{*,j}\lambda_0 + \big\|\Theta^*\big\|_1 \lambda_0 \Big), \tag{12}$$

$$\|\widehat{\Omega} - \Omega^*\|_2 \leq \|\widehat{\Omega} - \Omega^*\|_1 \leq C\Big(\max_{j\leq p}(\Omega^*_{jj})^{1/2} s_{*,j}\lambda_0 + \|\Omega^*\|_1 \lambda_0 \Big), \tag{13}$$

with large probability, where C is a constant depending on $\{c_0, c_*, A\}$ only. Moreover, the term $\|\Theta^*\|_1\lambda_0$ in (12) can be replaced by

$$\max_{j\leq p} \|\Theta^*_{*,j}\|_1 s_{*,j}\lambda_0^2 + \tau_n(\Theta^*), \tag{14}$$

where $\tau_n(M) = \inf\{\tau : \sum_j \exp(-n\tau^2/\|M_{*,j}\|_1^2) \leq 1/e\}$ for a matrix M.

Theorem 2 implies Theorem 1 due to $s_{*,j} \leq d-1$, $1/D_{jj} \leq \Theta^*_{jj} \leq \|\Theta^*\|_2$, $\|\Theta^*\|_1 \leq d\max_j \Theta^*_{jj}$ and similar inequalities for Ω^*. We note that $B_j = S_j$ in (11) gives the largest c_* and thus the sharpest error bounds in Theorem 2. In Section 7, we give an example to demonstrate the advantage of this theorem.

In a 2011 arXiv version of this paper (http://arxiv.org/pdf/1202.2723v1.pdf), we are able to demonstrate good numerical performance of the scaled Lasso estimator with the universal penalty level $\lambda_0 = \sqrt{2(\log p)/n}$, compared with some existing methods, but not the larger penalty level $\lambda_0 > \sqrt{4(\log p)/n}$ in Theorems 1 and 2. Since a main advantage of our proposal is automatic

selection of the penalty level without resorting to cross validation, a question arises as to whether a theory can be developed for a smaller penalty level to match the choice in a demonstration of good performance of the scaled Lasso in our simulation experiments.

We are able to provide an affirmative answer in this version of the paper by proving a higher concentration of the error of the scaled Lasso at a smaller penalty level as follows. Let $L_n(t)$ be the $N(0,1/n)$ quantile function satisfying

$$P\big\{N(0,1) > n^{1/2}L_n(t)\big\} = t.$$

Our earlier analysis is based on existing oracle inequalities of the Lasso which holds with probability $1-2\varepsilon$ when the inner product of design vectors and noise are bounded by their ε/p and $1-\varepsilon/p$ quantiles. Application of the union bound in p applications of the Lasso requires a threshold level $\lambda_{*,0} = L_n(\varepsilon/p^2)$ with a small $\varepsilon > 0$, which matches $\sqrt{4(\log p)/n}$ with $\varepsilon \asymp 1/\sqrt{\log p}$ in Theorems 1 and 2. Our new analysis of the scaled Lasso allows a threshold level

$$\lambda_{*,0} = L_{n-3/2}(k/p)$$

with $k \asymp s\log(p/s)$, where $s = 1+\max_j s_{*,j}$. More precisely, we require a penalty level $\lambda_0 \geq A\lambda_{*,0}$ with a constant A satisfying

$$A-1 > A_1 \geq \max_j \Big\{ \Big[\frac{e^{1/(4n-6)^2}4k}{m_j(L^4+2L^2)}\Big]^{1/2} + \frac{e^{1/(4n-6)^2}}{L\sqrt{2\pi}}\sqrt{\psi_j} + \frac{L_1(\varepsilon/p)}{L}\sqrt{\psi_j}\Big\}, \tag{15}$$

where $L = L_1(k/p)$, $s_{*,j} \leq m_j \leq \min(|B_j|, C_0 s_{*,j})$ with the $s_{*,j}$ and B_j in (10) and (11), and $\psi_j = \kappa_+(m_j; R^*_{-j,-j})/m_j + L_n(5\varepsilon/p^2)$ with

$$\kappa_+(m;\Sigma) = \max_{\|u\|_0=m, \|u\|_2=1} u^T\Sigma u. \tag{16}$$

Theorem 3 *Let $\{\overline{\Sigma}, \Sigma^*, \Theta^*, \Omega^*\}$ be matrices as in Theorem 2, and $\widehat{\Theta}$ and $\widehat{\Omega}$ be the scaled Lasso estimators with a penalty level $\lambda_0 \geq A\lambda_{*,0}$ where $\lambda_{*,0} = L_{n-3/2}(k/p)$. Suppose (10) and (11) hold with certain $\{S_j, s_{*,j}, \varepsilon_0, B_j, c_*\}$, (15) holds with constants $\{A, A_1, C_0\}$ and certain integers m_j, and $P\{(1-\varepsilon_0)^2 \leq \chi^2_n/n \leq (1+\varepsilon_0)^2\} \leq \varepsilon/p$. Then, there exist constants c_0 depending on c_* only and C depending on $\{A, A_1, C_0, c_*, c_0\}$ only such that when $\max_j s_{*,j}\lambda_0 \leq c_0$, the conclusions of Theorem 2 hold with at least probability $1-6\varepsilon-2k\sum_j(p-1-|B_j|)/p$.*

The condition $\max_j s_{*,j}\lambda_0 \leq c_0$ on (10), which controls the capped ℓ_1 sparsity of the inverse correlation matrix, weakens the ℓ_0 sparsity condition $d\sqrt{(\log p)/n} \to 0$.

The extra condition on the upper sparse eigenvalue $\kappa_+(m; R^*_{-j,-j})$ in (15) is mild, since it only requires a small $\kappa_+(m;R^*)/m$ that is actually decreasing in m.

The invertibility condition (11) is used to regularize the design matrix in linear regression procedures. As we mentioned earlier, condition (11) holds if the spectrum norm of Ω^* is bounded by $1/c_*$. Since $(R^*)^{-1} = \Omega^* = (\mathrm{diag}\Sigma^*)^{1/2}\Theta^*(\mathrm{diag}\Sigma^*)^{1/2}$, it suffices to have

$$\|(R^*)^{-1}\|_2 \leq \max\Sigma^*_{jj}\|\Theta^*\|_2 \leq 1/c_*.$$

To achieve the convergence rate $d\sqrt{(\log p)/n}$, both Yuan (2010) and Cai et al. (2011) require conditions $\|\Theta^*\|_1 = O(1)$ and $\max\Sigma^*_{jj} = O(1)$. In comparison, the spectrum norm condition is not only

weaker than the ℓ_1 operator norm condition, but also more natural for the convergence in spectrum norm.

Our sharper theoretical results are consequences of using the scaled Lasso estimator (4) and its fast convergence rate in linear regression. In Sun and Zhang (2012), a convergence rate of order $s_*(\log p)/n$ was established for the scaled Lasso estimation of the noise level, compared with an oracle noise level as the moment estimator based on the noise vector. In the context of the column-by-column application of the scaled Lasso for precision matrix estimation, the results in Sun and Zhang (2012) can be written as

$$\Big|\frac{\sigma_j^*}{\widehat{\sigma}_j}-1\Big| \le C_1 s_{*,j}\lambda_0^2, \quad \sum_{k\neq j}\overline{\Sigma}_{kk}^{1/2}|\widehat{\beta}_{k,j}-\beta_{k,j}|\sqrt{\Theta^*_{jj}} \le C_2 s_{*,j}\lambda_0, \tag{17}$$

where $\sigma_j^* = \|X\beta_{*,j}\|_2/\sqrt{n}$. We note that $n(\sigma_j^*)^2\Theta^*_{jj}$ is a chi-square variable with n degrees of freedom when X has iid $N(0,\Sigma^*)$ rows. The oracle inequalities in (17) play a crucial role in our analysis of the proposed estimators for inverse matrices, as the following proposition attests.

Proposition 4 *Let Θ^* be a nonnegative definite target matrix, $\Sigma^* = (\Theta^*)^{-1}$, and $\beta = -\Theta^*(\mathrm{diag}\Theta^*)^{-1}$. Let $\widehat{\Theta}$ and $\widehat{\Omega}$ be defined as (7) and (9) based on certain $\widehat{\beta}$ and $\widehat{\sigma}_j$ satisfying (17). Suppose further that*

$$|\Theta^*_{jj}(\sigma_j^*)^2 - 1| \le C_0\lambda_0, \ \max_j |(\overline{\Sigma}_{jj}/\Sigma^*_{jj})^{-1/2} - 1| \le C_0\lambda_0, \tag{18}$$

and that $\max\{4C_0\lambda_0, 4\lambda_0, C_1 s_{*,j}\lambda_0\} \le 1$. *Then, (12) and (13) hold with a constant C depending on $\{C_0, C_2\}$ only. Moreover, if $n\Theta^*_{jj}(\sigma_j^*)^2 \sim \chi^2_n$, then the term $\lambda_0\|\Theta^*\|_1$ in (12) can be replaced by (14) with large probability.*

While the results in Sun and Zhang (2012) requires a penalty level $A\sqrt{(2/n)\log(p^2)}$ to allow simultaneous application of (17) for all $j \le p$ via the union bound in proving Theorem 2, Theorem 3 allows a smaller penalty level $\lambda_{*,0} = AL_{n-3/2}(k/p)$ with $A > 1$ and a potentially large $k \asymp s\log(p/s)$. This is based on new theoretical results for the Lasso and scaled Lasso developed in Section 4.

4. Linear Regression Revisited

This section provides certain new error bounds for the Lasso and scaled Lasso in the linear regression model. Compared with existing error bounds, the new results characterize the concentration of the estimation and prediction errors at fixed, smaller threshold levels. The new results also allow high correlation among certain nuisance design vectors.

Consider the linear regression model with standardized design and normal error:

$$y = X\beta + \varepsilon, \ \|x_j\|_2^2 = n, \ \varepsilon \sim N(0, \sigma^2 I_n).$$

Let $\lambda_{univ} = \sqrt{(2/n)\log p}$ be the universal penalty level (Donoho and Johnstone, 1994). For the estimation of β and variable selection, existing theoretical results with $p \gg n$ typically require a penalty level $\lambda = A\sigma\lambda_{univ}$, with $A > 1$, to guarantee rate optimality of regularized estimators. This includes the scaled Lasso with a jointly estimated σ. For the Dantzig selector (Candès and Tao, 2007), performance bounds have been established for $A = 1$.

It is well understood that $\sigma\lambda_{univ}$ in such theorems is a convenient probabilistic upper bound of $\|X^T\varepsilon/n\|_\infty$ for controlling the maximum gradient of the squared loss $\|y-Xb\|_2^2/(2n)$ at $b=\widehat{\beta}$. For $\lambda < \|X^T\varepsilon/n\|_\infty$, variable selection is known to be inconsistent for the Lasso and most other regularized estimates of β, and the analysis of such procedures become more complicated due to false selection. However, this does not preclude the possibility that such a smaller λ outperforms the theoretical $\lambda \geq \sigma\lambda_{univ}$ for the estimation of β or prediction.

In addition to theoretical studies, a large volume of numerical comparisons among regularized estimators exists in the literature. In such numerical studies, the choice of penalty level is typically delegated to computationally more expensive cross-validation methods. Since cross-validation aims to optimize prediction performance, it may lead to a smaller penalty level than $\lambda = \sigma\lambda_{univ}$. However, this gap between $\lambda \geq \sigma\lambda_{univ}$ in theoretical studies and the possible choice of $\lambda < \sigma\lambda_{univ}$ in numerical studies is largely ignored in the existing literature.

The purpose of this section is to provide rate optimal oracle inequalities for the Lasso and its scaled version, which hold with at least probability $1-\varepsilon/p$ for a reasonably small ε, at a fixed penalty level λ satisfying $P\{N(0,\sigma^2/n) > \lambda/A\} = k/p$, with a given $A > 1$ and potentially large k, up to $k/(2\log(p/k))^2 \asymp s_*$, where s_* is a complexity measure of β, for example, $s_* = \|\beta\|_0$.

When the (scaled) Lasso is simultaneously applied to p subproblems as in the case of matrix estimation, the new oracle inequalities allow the use of the union bound to uniformly control the estimation error in subproblems at the same penalty level.

Rate optimal oracle inequalities have been established for ℓ_1 and concave regularized estimators in Zhang (2010) and Ye and Zhang (2010) for penalty level $\lambda = A\sigma\sqrt{c^*(2/n)\log(p/(\varepsilon s_*))}$, where c^* is an upper sparse eigenvalue, $A > 1$ and $1-\varepsilon$ is the guaranteed probability for the oracle inequality to hold. The new oracle inequalities remove the factors c^* and ε from the penalty level, as long as $1/\varepsilon$ is polynomial in p. The penalty level $A\sigma\sqrt{(2/n)\log(p/(\varepsilon s))}$ has been considered for models of size s under ℓ_0 regularization (Birge and Massart, 2001, 2007; Bunea et al., 2007; Abramovich and Grinshtein, 2010).

To bound the effect of the noise when $\lambda < \|X^T\varepsilon/n\|_\infty$, we use a certain sparse ℓ_2 norm to control the excess of $X^T\varepsilon/n$ over a threshold level λ_*. The sparse ℓ_q norm was used in the analysis of regularized estimators before (Candès and Tao, 2007; Zhang and Huang, 2008; Zhang, 2009; Cai et al., 2010; Ye and Zhang, 2010; Zhang, 2010), but it was done without a formal definition of the quantity to the best of our knowledge. To avoid repeating existing calculation, we define the norm and its dual here and summarize their properties in a proposition.

For $1 \leq q \leq \infty$ and $t > 0$, the sparse ℓ_q norm and its dual are defined as

$$\|v\|_{(q,t)} = \max_{|B|<t+1} \|v_B\|_q, \ \|v\|_{(q,t)}^* = \max_{\|u\|_{(q,t)}\leq 1} u^T v.$$

The following proposition describes some of their basic properties.

Proposition 5 *Let $m \geq 1$ be an integer, $q' = q/(q-1)$ and $a_q = (1-1/q)/q^{1/(q-1)}$.*
(i) Properties of $\|\cdot\|_{(q,m)}$: $\|v\|_{(q,m)} \downarrow q$, $\|v\|_{(q,m)}/m^{1/q} \downarrow m$, $\|v\|_{(q,m)}/m^{1/q} \uparrow q$,

$$\|v\|_\infty = \|v\|_{(q,1)} \leq \|v\|_{(q,m)} \leq (\|v\|_q) \wedge (m^{1/q}\|v\|_\infty),$$

and $\|v\|_q^q \leq \|v\|_{(q,m)}^q + (a_q/m)^{q-1}\|v\|_1^q$.
(ii) Properties of $\|\cdot\|_{(q,m)}^*$: $m^{1/q}\|v\|_{(q,m)}^* \downarrow q$, and

$$\max\left(\|v\|_{q'}, m^{-1/q}\|v\|_1\right) \leq \|v\|_{(q,m)}^* \leq \min\left(\|v\|_{(q',m/a_q)} + m^{-1/q}\|v\|_1, \|v\|_1\right).$$

(iii) Let $\overline{\Sigma} = X^T X/n$ *and* $\kappa_+(m;M)$ *be the sparse eigenvalue in (16). Then,*

$$\|\overline{\Sigma} v\|_{(2,m)} \le \min\Big\{\kappa_+^{1/2}(m;\overline{\Sigma})\|\overline{\Sigma}^{1/2} v\|_2, \kappa_+(m;\overline{\Sigma})\|v\|_2\Big\}.$$

4.1 Lasso with Smaller Penalty: Analytical Bounds

The Lasso path is defined as an $\mathbb{R}^p$-valued function of $\lambda > 0$ as

$$\widehat{\beta}(\lambda) = \arg\min_b \Big\{\|y - Xb\|_2^2/(2n) + \lambda\|b\|_1\Big\}.$$

For threshold levels $\lambda_* > 0$, we consider β satisfying the following complexity bound,

$$|S| + \sum_{j\notin S} |\beta_j|/\lambda_* \le s_* \tag{19}$$

with a certain $S \subset \{1,\ldots,p\}$. This includes the ℓ_0 sparsity condition $\|\beta\|_0 = s_*$ with $S = \mathrm{supp}(\beta)$ and allows $\|\beta\|_0$ to far exceed s_* with many small $|\beta_j|$.

The sparse ℓ_2 norm of a soft-thresholded vector v, at threshold level λ_* in (19), is

$$\zeta_{(2,m)}(v,\lambda_*) = \|(|v| - \lambda_*)_+\|_{(2,m)} = \max_{|J|\le m}\Big\{\sum_{j\in J}(|v_j| - \lambda_*)_+^2\Big\}^{1/2}. \tag{20}$$

Let $B \subseteq \{1,\ldots,p\}$ and

$$z = (z_1,\ldots,z_p)^T = X^T\varepsilon/n.$$

We bound the effect of the excess of the noise over λ_* under the condition

$$\|z_{B^c}\|_\infty \le \lambda_*,\ \zeta_{(2,m)}(z_B,\lambda_*) \le A_1 m^{1/2}\lambda_*, \tag{21}$$

for some $A_1 \ge 0$. We prove that when $\lambda \ge A\lambda_*$ with $A > 1 + A_1$ and (21) holds, a scaled version of $\widehat{\beta}(\lambda) - \beta$ belongs to a set $\mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, s_* - |S|)$, where

$$\begin{aligned} &\mathscr{U}(\Sigma, S, B; A, A_1, m, m_1) \\ = \ &\Big\{u : u^T\Sigma u + (A-1)\|u_{S^c}\|_1 \le (A+1)\|u_S\|_1 + A_1 m^{1/2}\|u_B\|_{(2,m)}^* + 2Am_1\Big\}. \end{aligned} \tag{22}$$

This leads to the definition of

$$M_{pred}^* = \sup\Big\{\frac{u^T\overline{\Sigma}u/A^2}{m_1 + |S|} : u \in \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1)\Big\} \tag{23}$$

as a constant factor for the prediction error of the Lasso and

$$M_q^* = \sup\Big\{\frac{\|u\|_q/A}{(m_1 + |S|)^{1/q}} : u \in \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1)\Big\} \tag{24}$$

for the ℓ_q estimation error of the Lasso.

The following theorem provides analytic error bounds for the Lasso prediction and estimation under the sparse ℓ_2 norm condition (21) on the noise. This is different from existing analyses of the Lasso based on the ℓ_∞ noise bound $\|X^T\varepsilon/n\|_\infty \le \lambda_*$. In the case of Gaussian error, (21) allows a fixed threshold level $\lambda_* = \sigma\sqrt{(2/n)\log(p/m)}$ to uniformly control the error of p applications of the Lasso for the estimation of a precision matrix. When $m \asymp s_*$ and $\sigma\sqrt{(2/n)\log(p/m)} \ll \sigma\sqrt{(2/n)\log p}$, using such smaller λ_* is necessary for achieving error bounds with the sharper rate corresponding to $\sigma\sqrt{(2/n)\log(p/m)}$.

Theorem 6 *Suppose (19) holds with certain $\{S, s_*, \lambda_*\}$. Let $A > 1$, $\widehat{\beta} = \widehat{\beta}(\lambda)$ be the Lasso estimator with penalty level $\lambda \geq A\lambda_*$, $h = \widehat{\beta} - \beta$, and $m_1 = s_* - |S|$. If (21) holds with $A_1 \geq 0$, a positive integer m and $B \subseteq \{1, \ldots, p\}$, then*

$$\|Xh\|_2^2/n \leq M_{pred}^* s_* \lambda^2, \quad \|h\|_q \leq M_q^* s_*^{1/q} \lambda. \tag{25}$$

Remark 7 *Theorem 6 covers $\|X^T\varepsilon/n\|_\infty \leq \lambda_*$ as a special case with $A_1 = 0$. In this case, the set (22) does not depend on $\{m, B\}$. For $A_1 = 0$ and $|S| = s_*$ ($m_1 = 0$), (22) contains all vectors satisfying a basic inequality $u^T\Sigma u + (A-1)\|u_{S^c}\|_1 \leq (A+1)\|u_S\|_1$ (Bickel et al., 2009; van de Geer and Bühlmann, 2009; Ye and Zhang, 2010) and Theorem 6 still holds when (22) is replaced by the smaller*

$$\mathscr{U}_-(\Sigma, S, A) = \Big\{ u : \|\overline{\Sigma}_{S,*} u\|_\infty \leq A+1, u_j \overline{\Sigma}_{j,*} u \leq -|u_j|(A-1) \ \forall j \notin S \Big\}.$$

Thus, in what follows, we always treat $\mathscr{U}(\Sigma, S, B; A, 0, m, 0)$ as $\mathscr{U}_-(\Sigma, S, A)$ when $A_1 = 0$ and $|S| = s_*$. This yields smaller constants $\{M_{pred}^*, M_q^*\}$ in (23) and (24).

The purpose of including a choice B in (21) is to achieve bounded $\{M_{pred}^*, M_1^*\}$ in the presence of some highly correlated design vectors outside $S \cup B$ when $\overline{\Sigma}_{S\cup B,(S\cup B)^c}$ is small. Since $\|u_B\|_{(2,m)}^*$ is increasing in B, a larger B leads to a larger set (22) and larger $\{M_{pred}^*, M_q^*\}$. However, (21) with smaller B typically requires larger λ_*. Fortunately, the difference in the required λ_* in (21) is of smaller order than λ_* between the largest $B = \{1, \ldots, p\}$ and smaller B with $|B^c| \leq p/m$. We discuss the relationship between $\{M_{pred}^*, M_q^*\}$ and existing conditions on the design in the next section, along with some simple upper bounds for $\{M_{pred}^*, M_1^*, M_2^*\}$.

4.2 Scaled Lasso with Smaller Penalty: Analytical Bounds

The scaled Lasso estimator is defined as

$$\{\widehat{\beta}, \widehat{\sigma}\} = \arg\min_{b,\sigma} \Big\{ \|y - Xb\|_2^2/(2n\sigma) + \lambda_0 \|b\|_1 + \sigma/2 \Big\}, \tag{26}$$

where $\lambda_0 > 0$ is a scale-free penalty level. In this section, we describe the implication of Theorem 6 on the scaled Lasso.

A scaled version of (19) is

$$|S| + \sum_{j \notin S} |\beta_j|/(\sigma^* \lambda_{*,0}) = s_{*,0} \leq s_*, \tag{27}$$

where $\sigma^* = \|\varepsilon\|_2/\sqrt{n}$ is an oracle estimate of the noise level and $\lambda_{*,0} > 0$ is a scaled threshold level. This holds automatically under (19) when $S \supseteq \mathrm{supp}(\beta)$. When $\beta_{S^c} \neq 0$, (27) can be viewed as an event of large probability. When

$$|S| + (1-\varepsilon_0)^{-1} \sum_{j \notin S} \frac{|\beta_j|}{\sigma \lambda_{*,0}} \leq s_* \tag{28}$$

and $\varepsilon \sim N(0, \sigma I_n)$, $P\{s_{*,0} \leq s_*\} \geq P\{\chi_n^2/n \geq (1-\varepsilon_0)^2\} \to 1$ for fixed $\varepsilon_0 > 0$. Let

$$M_\sigma^* = \sup_{u \in \mathscr{U}} \left\{ \frac{u^T \overline{\Sigma} u}{s_* A^2} + \frac{2\|u\|_1}{s_* A^2} + \frac{2A_1 m^{1/2} \|u_B\|_{(2,m)}^*}{s_* A^2} \right\} \tag{29}$$

with $\mathscr{U} = \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1)$ in (22), as in (23) and (24). Set

$$\eta_* = M_\sigma^* A^2 \lambda_{*,0}^2 s_*,\ \lambda_0 \geq A\lambda_{*,0}/\sqrt{(1-\eta_*)_+},\ \eta_0 = M_\sigma^* \lambda_0^2 s_*.$$

Theorem 8 *Suppose $\eta_0 < 1$. Let $\{\widehat{\beta}, \widehat{\sigma}\}$ be the scaled Lasso estimator in (26), $\phi_1 = 1/\sqrt{1+\eta_0}$, $\phi_2 = 1/\sqrt{1-\eta_0}$, and $\sigma^* = \|\varepsilon\|_2/\sqrt{n}$. Suppose (21) holds with $\{z_B, \lambda_*\}$ replaced by $\{z_B/\sigma^*, \lambda_{*,0}\}$.*
(i) Let $h^* = (\widehat{\beta} - \beta)/\sigma^*$. Suppose (27) holds. Then,

$$\phi_1 < \widehat{\sigma}/\sigma^* < \phi_2,\ \|Xh^*\|_2^2/n < M_{pred}^* s_* (\phi_2 \lambda_0)^2,\ \|h^*\|_q < M_q^* s_* \phi_2 \lambda_0. \tag{30}$$

(ii) Let $h = \widehat{\beta} - \beta$. Suppose (28) holds and $1 - \varepsilon_0 \leq \sigma^*/\sigma \leq 1 + \varepsilon_0$. Then,

$$\begin{aligned} &(1-\varepsilon_0)\phi_1 < \widehat{\sigma}/\sigma < \phi_2(1+\varepsilon_0), \\ &\|Xh\|_2^2/n < (1+\varepsilon_0)^2 M_{pred}^* s_* (\sigma\phi_2\lambda_0)^2, \\ &\|h\|_q < (1+\varepsilon_0) M_q^* s_* \sigma \phi_2 \lambda_0. \end{aligned} \tag{31}$$

Compared with Theorem 6, Theorem 8 requires nearly identical conditions on the design X, the noise and penalty level under proper scale. It essentially allows the substitution of $\{y, X, \beta\}$ by $\{y/\sigma^*, X, \beta/\sigma^*\}$ when η_0 is small.

Theorems 6 and 8 require an upper bound (21) for the sparse ℓ_2 norm of the excess noise as well as upper bounds for the constant factors $\{M_{pred}^*, M_q^*, M_\sigma^*\}$ in (23), (24) and (29). Probabilistic upper bounds for the noise and consequences of their combination with Theorems 6 and 8 are discussed in Section 4.3. We use the rest of this subsection to discuss $\{M_{pred}^*, M_q^*, M_\sigma^*\}$.

Existing analyses of the Lasso and Dantzig selector can be to used find upper bounds for $\{M_{pred}^*, M_q^*, M_\sigma^*\}$ via the sparse eigenvalues (Candès and Tao, 2005, 2007; Zhang and Huang, 2008; Zhang, 2009; Cai et al., 2010; Zhang, 2010; Ye and Zhang, 2010). In the simpler case $A_1 = m_1 = 0$, shaper bounds can be obtained using the compatibility factor (van de Geer, 2007; van de Geer and Bühlmann, 2009), the restricted eigenvalue (Bickel et al., 2009; Koltchinskii, 2009), or the cone invertibility factors (Ye and Zhang, 2010; Zhang and Zhang, 2012). Detailed discussions can be found in van de Geer and Bühlmann (2009), Ye and Zhang (2010) and Zhang and Zhang (2012) among others. The main difference here is the possibility of excluding some highly correlated vectors from B in the case of $A_1 > 0$. The following lemma provide some simple bounds used in our analysis of the scaled Lasso estimation of the precision matrix.

Lemma 9 *Let $\{M_{pred}^*, M_q^*, M_\sigma^*\}$ be as in (23), (24) and (29) with the vector class $\mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1)$ in (22). Suppose that for a nonnegative-definite matrix Σ, $\max_j \|\overline{\Sigma}_{j,*} - \Sigma_{j,*}\|_\infty \leq \lambda^*$ and $c_*\|u_{S\cup B}\|_2^2 \leq u^T\Sigma u$ for $u \in \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1)$. Suppose further that $\lambda^*\{(s_* \vee m)/c_*\}(2A + A_1)^2 \leq (A - A_1 - 1)_+^2/2$. Then,*

$$M_{pred}^* + M_1^*\Big(1 - \frac{A_1+1}{A}\Big) \leq \max\Big\{\frac{4 \vee (4m/s_*)}{c_*(2 + A_1/A)^{-2}}, \frac{c_*(1 - |S|/s_*)}{A^2}\Big\} \tag{32}$$

and

$$M_\sigma^* \leq \Big(1 + \frac{2A_1}{c_* A}\Big) M_{pred}^* + 2(1+A_1)\frac{M_1^*}{A} + \frac{A_1 m}{A s_*} + \frac{2A_1}{A^3}\Big(1 - \frac{|S|}{s_*}\Big). \tag{33}$$

Moreover, if in addition $B = \{1, \ldots, p\}$ then

$$M_2^* \leq (2/c_*) M_{pred}^* + 2(1 - |S|/s_*)/(A^2). \tag{34}$$

The main condition of Lemma 9,

$$c_* \leq \inf\left\{ \frac{u^T \Sigma u}{\|u_{S\cup B}\|_2^2} : u \in \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1) \right\}, \tag{35}$$

can be viewed as a restricted eigenvalue condition (Bickel et al., 2009) on a population version of the Gram matrix. However, one may also pick the sample version $\Sigma = \overline{\Sigma}$ with $\lambda^* = 0$. Let $\{A, A_1\}$ be fixed constants satisfying $A_1 < A - 1$. Lemma 9 asserts that the factors $\{M^*_{pred}, M^*_1, M^*_\sigma\}$ can be all treated as constants when $1/c_*$ and m/s_* are bounded and $\lambda^*(s_* \vee m)/c_*$ is smaller than a certain constant. Moreover, M^*_2 can be also treated as a constant when (35) holds for $B = \{1, \ldots, p\}$.

4.3 Probabilistic Error Bounds

Theorems 6 and 8 provides analytical error bounds based on the size of the excess noise over a given threshold. Here we provide probabilistic upper bounds for the excess noise and describe their implications in combination with Theorems 6 and 8. We use the following notation:

$$L_n(t) = n^{-1/2}\Phi^{-1}(1-t), \tag{36}$$

where $\Phi^{-1}(t)$ is the standard normal quantile function.

Proposition 10 *Let $\zeta_{(2,m)}(v, \lambda_*)$ be as in (20) and $\kappa_+(m) = \kappa_+(m; \overline{\Sigma})$ as in (16) with $\overline{\Sigma} = X^T X/n$. Suppose $\varepsilon \sim N(0, \sigma^2 I_n)$ and $\|x_j\|_2^2 = n$. Let $k > 0$.*
(i) Let $z = X^T\varepsilon/n$ and $\lambda_* = \sigma L_n(k/p)$. Then, $P\{\zeta_{(2,p)}(z, \lambda_*) > 0\} \leq 2k$, and

$$E\zeta^2_{(2,p)}(z, \lambda_*) \leq 4k\lambda_*^2/\{L_1^4(k/p) + 2L_1^2(k/p)\},$$
$$P\left\{ \zeta_{(2,m)}(z, \lambda_*) > E\zeta_{(2,p)}(z, \lambda_*) + \sigma L_n(\varepsilon)\sqrt{\kappa_+(m)} \right\} \leq \varepsilon. \tag{37}$$

(ii) Let $\sigma^* = \|\varepsilon\|_2/\sqrt{n}$, $z^* = z/\sigma^*$, $\lambda_{*,0} = L_{n-3/2}(k/p)$ and $\varepsilon_n = e^{1/(4n-6)^2} - 1$. Then, $P\{\zeta_{(2,p)}(z^*, \lambda_{*,0}) > 0\} \leq (1+\varepsilon_n)k$, $E\zeta^2_{(2,p)}(z^*, \lambda_{*,0}) \leq (1+\varepsilon_n)4k\lambda^2_{*,0}/\{L_1^4(k/p) + 2L_1^2(k/p)\}$, and

$$P\left\{ \zeta_{(2,m)}(z^*, \lambda_{*,0}) > \mu_{(2,m)} + L_{n-3/2}(\varepsilon)\sqrt{\kappa_+(m)} \right\} \leq (1+\varepsilon_n)\varepsilon, \tag{38}$$

where $\mu_{(2,m)}$ is the median of $\zeta_{(2,m)}(z^*, \lambda_{*,0})$. Moreover,

$$\mu_{(2,m)} \leq E\zeta_{(2,p)}(z^*, \lambda_{*,0}) + (1+\varepsilon_n)\{\lambda_{*,0}/L_1(k/p)\}\sqrt{\kappa_+(m)/(2\pi)}. \tag{39}$$

We describe consequences of combining Proposition 10 with Theorems 6 and 8 in three theorems, respectively using the probability of no excess noise over the threshold, the Markov inequality with the second moment, and the concentration bound on the excess noise.

Theorem 11 *Let $0 < \varepsilon < p$. Suppose $\varepsilon \sim N(0, \sigma^2 I_n)$.*
(i) Let the notation be as in Theorem 6 and (36) with $A_1 = 0$ and $\lambda_* = \sigma L_n(\varepsilon/p^2)$. If (19) holds, then (25) holds with at least probability $1 - 2\varepsilon/p$.
(ii) Let the notation be as in Theorem 8 and (36) with $A_1 = 0$ and $\lambda_{*,0} = L_{n-3/2}(\varepsilon/p^2)$. If (28) holds with $P\{(1-\varepsilon_0)^2 \leq \chi^2_n/n \leq (1+\varepsilon_0)^2\} \leq \varepsilon/p$, then (30) and (31) hold with at least probability $1 - 3\varepsilon/p$.

For a single application of the Lasso or scaled Lasso, $\varepsilon/p = o(1)$ guarantees $\|z\|_\infty \le \lambda_*$ in Theorem 11 (i) and $\|z^*\|_\infty \le \lambda_{*,0}$ in Theorem 11 (ii) with high probability. The threshold levels are $\lambda_*/\sigma \approx \lambda_{*,0} \approx \lambda_{univ} = \sqrt{(2/n)\log p}$, as typically considered in the literature. In numerical experiments, this often produces nearly optimal results although the threshold level may still be somewhat higher than optimal for the prediction and estimation of β. However, if we use the union bound to guarantee the simultaneous validity of the oracle inequalities in p applications of the scaled Lasso in the estimation of individual columns of a precision matrix, Theorem 11 requires $\varepsilon = o(1)$, or equivalently a significantly higher threshold level $\lambda_{*,0} \approx \sqrt{(4/n)\log p}$. This higher $\lambda_{*,0}$, which does not change the theoretical results by much, may produce clearly suboptimal results in numerical experiments.

Theorem 12 *Let $k > 0$. Suppose $\varepsilon \sim N(0, \sigma^2 I_n)$.*
(i) Let the notation be as in Theorem 6 and Proposition 10, $\lambda_* = \sigma L_n(k/p)$, and $A - 1 > A_1 \ge \sqrt{4k/(\varepsilon m(L_1^4(k/p) + 2L_1^2(k/p)))}$. If (19) holds, then (25) holds with at least probability $1 - \varepsilon - 2|B^c|k/p$.
(ii) Let the notation be as in Theorem 8 and Proposition 10, $\lambda_{*,0} = L_{n-3/2}(k/p)$, $\varepsilon_n = e^{1/(4n-6)^2} - 1$, and $A - 1 > A_1 \ge \sqrt{(1+\varepsilon_n)4k/(\varepsilon m(L_1^4(k/p) + 2L_1^2(k/p)))}$. If (28) holds with $P\{(1-\varepsilon_0)^2 \le \chi_n^2/n \le (1+\varepsilon_0)^2\} \le \varepsilon$, then (30) and (31) hold with at least probability $1 - 2\varepsilon - 2|B^c|k/p$.

Theorem 12 uses the upper bounds for $E\zeta_{(2,p)}^2(z, \lambda_*)$ and $E\zeta_{(2,p)}^2(z^*, \lambda_{*,0})$ to verify (21). Since $L_n(k/p) \approx \sqrt{(2/n)\log(p/k)}$, it allows smaller threshold levels λ_* and $\lambda_{*,0}$ as long as $k/(\varepsilon m(L_1^4(k/p) + 2L_1^2(k/p)))$ is small. However, it does not allow $\varepsilon \le 1/p$ for using the union bound in p applications of the Lasso in precision matrix estimation.

Theorem 13 *Let $k > 0$. Suppose $\varepsilon \sim N(0, \sigma^2 I_n)$.*
(i) Let the notation be as in Theorem 6 and Proposition 10, $\lambda_* = \sigma L_n(k/p)$, and

$$A - 1 > A_1 \ge \Big(\frac{4k/m}{L_1^4(k/p) + 2L_1^2(k/p)}\Big)^{1/2} + \frac{L_1(\varepsilon/p)}{L_1(k/p)}\Big(\frac{\kappa_+(m)}{m}\Big)^{1/2}.$$

If (19) holds, then (25) holds with at least probability $1 - \varepsilon/p - 2|B^c|k/p$.
(ii) Let the notation be as in Theorem 8 and Proposition 10, $\lambda_{*,0} = L_{n-3/2}(k/p)$, $\varepsilon_n = e^{1/(4n-6)^2} - 1$, and

$$A - 1 > A_1 \ge \Big(\frac{(1+\varepsilon_n)4k/m}{L_1^4(k/p) + 2L_1^2(k/p)}\Big)^{1/2} + \Big(\frac{L_1(\varepsilon/p)}{L_1(k/p)} + \frac{1+\varepsilon_n}{L_1(k/p)\sqrt{2\pi}}\Big)\Big(\frac{\kappa_+(m)}{m}\Big)^{1/2}.$$

If (28) holds with $P\{(1-\varepsilon_0)^2 \le \chi_n^2/n \le (1+\varepsilon_0)^2\} \le \varepsilon/p$, then (30) and (31) hold with at least probability $1 - 2\varepsilon/p - 2|B^c|k/p$.

Theorem 13 uses concentration inequalities (37) and (38) to verify (21). Let $B = \{1, \ldots, p\}$ and $L_n(t)$ be as in (36). By guaranteeing the validity of the oracle inequalities with $1 - \varepsilon/p$ probability, with a reasonably small ε, Theorem 13 justifies the use of a fixed smaller threshold level $\lambda_{*,0} =$

$L_{n-3/2}(k/p) \approx \sqrt{(2/n)\log(p/k)}$ in p applications of the scaled Lasso to estimate columns of a precision matrix.

Since $L_1(\varepsilon/p) \approx L_1(k/p)$ typically holds, Theorem 13 only requires $(k/m)/(L_1^4(k/p)+2L_1^2(k/p))$ and $\kappa_+(m)/m$ be smaller than a fixed small constant. This condition relies on the upper sparse eigenvalue only in a mild way since $\kappa_+(m)/m$ is decreasing in m and $\kappa_+(m)/m \le 1/m + (1-1/m)\max_{j\neq k}|x_j^T x_k/n|$ (Zhang and Huang, 2008).

For $k \asymp m$ and $\log(p/k) \asymp \log(p/(s_* \vee 1))$, Theorem 13 provides prediction and ℓ_q error bounds of the orders $\sigma^2(s_* \vee 1)\lambda_{*,0}^2 \approx \sigma^2((s_* \vee 1)/n)2\log(p/(s_* \vee 1))$ and $\sigma(s_* \vee 1)^{1/q}\lambda_{*,0}$ respectively. For $\log(p/n) \ll \log n$, this could be of smaller order than the error bounds with $\lambda_{*,0} \approx \lambda_{univ} = \sqrt{(2/n)\log p}$.

Theorem 13 suggests the use of a penalty level satisfying $\lambda/\sigma = \lambda_0 = AL_n(k/p) \approx A\sqrt{(2/n)\log(p/k)}$ with $1 < A \le \sqrt{2}$ and a real solution of $k = L_1^4(k/p) + 2L_1^2(k/p)$. This is conservative since the constraint on A in the theorem is valid with a moderate $m = O(s_* + 1)$. For p applications of the scaled Lasso in the estimation of precision matrix, this also provides a more practical penalty level compared with $A'L_n(\varepsilon/p^2) \approx A'\sqrt{(4/n)\log(p/\varepsilon^{1/2})}$, $A' > 1$ and $\varepsilon \ll 1$, based on existing results and Theorem 11. In our simulation study, we use $\lambda_0 = \sqrt{2}L_n(k/p)$ with $k = L_1^4(k/p) + 2L_1^2(k/p)$.

4.4 A Lower Performance Bound

It is well understood that in the class of β satisfying the sparsity condition (19), $s_*\sigma^2 L_n^2(s_*/p)$ and $s_*^{1/q}\sigma L_n(s_*/p)$ are respectively lower bounds for the rates of minimax prediction and ℓ_q estimation error (Ye and Zhang, 2010; Raskutti et al., 2011). This can be achieved by the Lasso with $\lambda_* = \sigma L_n(m/p)$, $m \asymp s_*$, or scaled Lasso with $\lambda_{*,0} = L_{n-3/2}(m/p)$. The following proposition asserts that for each fixed β, the minimax error rate cannot be achieved by regularizing the gradient with a threshold level of smaller order.

Proposition 14 *Let $y = X\beta + \varepsilon$, $\widetilde{\beta}(\lambda)$ satisfy $\|X^T(y - X\widetilde{\beta}(\lambda))/n\|_\infty \le \lambda$, and $\widetilde{h}(\lambda) = \widetilde{\beta}(\lambda) - \beta$. Let $\overline{\Sigma} = X^T X/n$ and $\kappa_+(m;\cdot)$ be as in (16).*
(i) If $\|X^T\varepsilon/n\|_{(2,k)} \ge k^{1/2}\lambda_* > 0$, then for all $A > 1$

$$\inf_{\lambda \le \lambda_*/A} \min\Big\{ \frac{\|X\widetilde{h}(\lambda)\|^2/n}{\kappa_+^{-1}(k;\overline{\Sigma})}, \frac{\|\widetilde{h}(\lambda)\|_2^2}{\kappa_+^{-2}(k;\overline{\Sigma})} \Big\} \ge (1-1/A)^2 k\lambda_*^2. \tag{40}$$

(ii) Let $\sigma^* = \|\varepsilon\|_2/\sqrt{n}$ and $N_k = \#\{j : |x_j^T\varepsilon|/(n\sigma^*) \ge \widetilde{L}_n(k/p)\}$ with $\widetilde{L}_n(t) = L_n(t) - n^{-1/2}$. Suppose X has iid $N(0,\Sigma)$ rows, $\mathrm{diag}(\Sigma) = I_p$, and $2k - 4\|\Sigma\|_2 \ge \big(\sqrt{k-1} + \sqrt{2\|\Sigma\|_2\log(1/\varepsilon)}\big)^2$. Then, $P\{N_k \ge k\} \ge 1-\varepsilon$ and

$$P\Big\{ \|X^T\varepsilon/n\|_{(q,k)} \ge \sigma^* k^{1/q}\sigma\widetilde{L}_n(k/p) \Big\} \ge 1-\varepsilon. \tag{41}$$

Consequently, there exist numerical constants c_1 and c_2 such that

$$P\Big\{ \inf_{\lambda \le c_1\sigma L_n(k/p)} \min\Big(\frac{\|X\widetilde{h}(\lambda)\|^2/n}{\kappa_+^{-1}(k;\overline{\Sigma})}, \frac{\|\widetilde{h}(\lambda)\|_2^2}{\kappa_+^{-2}(k;\overline{\Sigma})} \Big) \ge c_2\sigma^2 k L_n^2(k/p) \Big\} \ge 1 - \varepsilon - e^{-n/9}.$$

It follows from Proposition 14 (ii) that the prediction and ℓ_2 estimation error is of no smaller order than $k\sigma^2 L_n^2(k/p)$ for all $\lambda \le c_1\sigma L_n(k/p)$. This rate is suboptimal when $k\log(p/k) \gg s_*\log(p/s_*)$.

5. Estimation after Model Selection

We have presented theoretical properties of the scaled Lasso for linear regression and precision matrix estimation. After model selection, the least squares estimator is often used to remove bias of regularized estimators. The usefulness of this technique after the scaled Lasso was demonstrated in Sun and Zhang (2012), along with its theoretical justification. In this section, we extend the theory to smaller threshold level and to the estimation of precision matrix.

In linear regression, the least squares estimator β and the corresponding estimate of σ in the model selected by a regularized estimator $\widehat{\beta}$ are given by

$$\overline{\beta} = \arg\min_b \left\{ \|y - Xb\|_2^2 : \mathrm{supp}(b) \subseteq \widehat{S} \right\}, \quad \overline{\sigma} = \left\|y - X\overline{\beta}\right\|_2 / \sqrt{n}, \tag{42}$$

where $\widehat{S} = \mathrm{supp}(\widehat{\beta})$. To study the performance of (42), we define sparse eigenvalues relative to a support set S as follows:

$$\kappa_-^*(m^*, S; \Sigma) = \min_{J \supseteq S, |J \setminus S| \le m^*} \min_{\|u_J\|_2 = 1} u_J^T \Sigma_{J,J} u_J,$$
$$\kappa_+^*(m^*, S; \Sigma) = \min_{J \cap S = \emptyset, |J| \le m^*} \max_{\|u_J\|_2 = 1} u_J^T \Sigma_{J,J} u_J.$$

It is proved in Sun and Zhang (2012) that $\{\overline{\beta}, \overline{\sigma}\}$ satisfies prediction and estimation error bounds of the same order as those for the scaled Lasso (26) under some extra conditions on $\kappa_\pm^*(m^*, S; \overline{\Sigma})$. The extra condition on $\kappa_+^*(m^*, S; \overline{\Sigma})$ is used to derive an upper bound for the false positive $|\widehat{S} \setminus S|$, and then the extra condition on $\kappa_-^*(m^*, S; \overline{\Sigma})$ is used to invert $X_{S \cup \widehat{S}}$. The following theorem extends the result to the smaller threshold level $\lambda_{*,0} = L_{n-3/2}(k/p)$ in Theorem 13 (ii). Let

$$M_{lse}^* = \frac{\left[\left\{ |S| + \left(\sqrt{m^*} + \sqrt{2m^* \log(ep/m^*)} \right)^2 \right\}^{1/2} + L_1(\varepsilon/p) \right]^2}{s_* \log(p/s_*)}.$$

Theorem 15 *Let $(\widehat{\beta}, \widehat{\sigma})$ be the scaled lasso estimator in (26) and $(\overline{\beta}, \overline{\sigma})$ be the least squares estimator (42) in the selected model $\widehat{S} = \mathrm{supp}(\widehat{\beta})$. Let the notation be as in Theorem 13 (ii) and $m^* > m$ be an integer satisfying $s_* M_{pred}^* / \{(1-\xi_1)(1-1/A)\}^2 \le m^* / \kappa_+^*(m^*, S)$. Suppose $\beta_{S^c} = 0$ and (21) holds with $\{z_B, \lambda_*\}$ replaced by $\{z_B^*, \lambda_{*,0}\}$. Then,*

$$|\widehat{S} \setminus S| \le m^* \tag{43}$$

with at least probability $1 - 2\varepsilon/p - 2|B^c|k/p$. Moreover,

$$\begin{aligned} & (\sigma^*)^2 - \sigma^2 M_{lse}^* (s_*/n) \log(p/s_*) \\ \le\ & \overline{\sigma}^2 \\ \le\ & \widehat{\sigma}^2, \\ & \kappa_-^*(m^* - 1, S) \|\overline{h}\|_2^2 \\ \le\ & \|X\overline{h}\|_2^2 / n \\ \le\ & (1+\varepsilon_0)^2 M_{pred}^* s_* (\sigma \phi_2 \lambda_0)^2 + \sigma^2 M_{lse}^* (s_*/n) \log(p/s_*), \end{aligned} \tag{44}$$

with at least probability $1 - 3\varepsilon/p - 2|B^c|k/p$, where $\overline{h} = \overline{\beta} - \beta$.

Theorem 15 asserts that when $k \vee m^* \asymp s_*$, the least squares estimator $\{\overline{\beta}, \overline{\sigma}\}$ after the scaled Lasso selection enjoys estimation and prediction properties comparable to that of the scaled Lasso:

$$\lambda_0^{-2}\Big\{\big|\overline{\sigma}/\sigma^* - 1\big| + \big\|\overline{\beta} - \beta\big\|_2^2 + \big\|X\overline{\beta} - X\beta\big\|_2^2/n\Big\} + \|\overline{\beta}\|_0 = O_P(1)s_*.$$

Now we apply this method for precision matrix estimation. Let $\widehat{\beta}$ be as in (4) and define $\overline{\beta}$ and $\overline{\sigma}$ as follows:

$$\begin{aligned} \overline{\beta}_{*,j} &= \arg\min_b \Big\{ \|Xb\|_2^2 : b_j = -1, \mathrm{supp}(b) \subseteq \mathrm{supp}(\widehat{\beta}_{*,j}) \Big\}, \\ \overline{\sigma}_j &= \big\|X\overline{\beta}_{*,j}\big\|_2/\sqrt{n}. \end{aligned} \tag{45}$$

We define $\widetilde{\Theta}^{\mathrm{LSE}}$ and $\widehat{\Theta}^{\mathrm{LSE}}$ as in (7) and (9) with $\overline{\beta}$ and $\overline{\sigma}$ in place of $\widehat{\beta}$ and $\widehat{\sigma}$.

Under an additional condition on the upper sparse eigenvalue, Theorem 15 is parallel to Theorem 8 (ii), and the theoretical results in Sun and Zhang (2012) are parallel to Theorem 6 (ii). These results can be used to verify the condition (17), so that Proposition 4 also applies to (45) with the extra upper sparse eigenvalue condition on the population correlation matrix R^*. We formally state this result as a corollary.

Corollary 16 *Under the additional condition $\|R^*\|_2 = O(1)$ on the population correlation matrix R^*, Theorems 2 and 3 are applicable to the estimator $\widehat{\Theta}^{LSE}$ and the corresponding estimator for $\Omega^* = (R^*)^{-1}$ with possibly different numerical constants.*

6. Numerical Study

In this section, we present some numerical comparison between the proposed and existing methods. In addition to the proposed estimator (7) and (9) based on the scaled Lasso (4) and the least squares estimation after the scale Lasso (45), the graphical Lasso and CLIME are considered. The following three models are considered. Models 1 and 2 have been considered in Cai et al. (2011), while Model 2 in Rothman et al. (2008).

- Model 1: $\Theta_{ij} = 0.6^{|i-j|}$.
- Model 2: Let $\Theta = B + \delta I$, where each off-diagonal entry in B is generated independently and equals to 0.5 with probability 0.1 or 0 with probability 0.9. The constant δ is chosen such that the condition number of Θ^* is p. Finally, we rescale the matrix Θ^* to the unit in diagonal.
- Model 3: The diagonal of the target matrix has unequal values. $\Theta = D^{1/2}\Omega D^{1/2}$, where $\Omega_{ij} = 0.6^{|i-j|}$ and D is a diagonal matrix with diagonal elements $d_{ii} = (4i + p - 5)/\{5(p-1)\}$.

Among the three models, Model 2 is the densest. For $p = 1000$, the capped ℓ_1 sparsity s_* is 8.84, 24.63, and 8.80 for three models respectively.

In each model, we generate a training sample of size 100 from a multivariate normal distribution with mean zero and covariance matrix $\Sigma = \Theta^{-1}$ and an independent sample of size 100 from the same distribution for validating the tuning parameter λ for the graphical Lasso and CLIME. The GLasso and CLIME estimators are computed based on training data with various λ's and we choose λ by minimizing likelihood loss $\{\mathrm{trace}(\overline{\Sigma}\widehat{\Theta}) - \log\det(\widehat{\Theta})\}$ on the validation sample. The scaled

Lasso estimators are computed based on the training sample alone with penalty level $\lambda_0 = AL_n(k/p)$, where $A = \sqrt{2}$ and k is the solution of $k = L_1^4(k/p) + 2L_1^2(k/p)$. The symmetrization step in Cai et al. (2011) is applied. We consider six different dimensions $p = 30, 60, 90, 150, 300, 1000$ and replicate 100 times in each setting. The CLIME estimators for $p = 300$ and $p = 1000$ are not computed due to computational costs.

Table 1 presents the mean and standard deviation of estimation errors based on 100 replications. The estimation error is measured by three matrix norms: the spectrum norm, the matrix ℓ_1 norm and the Frobenius norm. The scaled Lasso estimator, labeled as SLasso, outperforms the graphical Lasso (GLasso) in all cases except for the smaller $p \in \{30, 60, 90\}$ in the Frobenius loss in the denser Model 2. It also outperforms the CLIME in most cases, except for smaller p in sparser models ($p = 30$ in Model 1 and $p \in \{30, 60\}$ in Model 3). The least squares estimator after the scaled Lasso selection outperforms all estimators by large margin in the spectrum and Frobenius losses in Models 1 and 3, but in general underperforms in the ℓ_1 operator norm and in Model 2. It seems that post processing by the least squares method is a somewhat aggressive procedure for bias correction. It performs well in sparse models, where variable selection is easier, but may not perform very well in denser models.

Both the scaled Lasso and the CLIME are resulting from sparse linear regression solutions. A main advantage of the scaled Lasso over the CLIME is adaptive choice of the penalty level for the estimation of each column of the precision matrix. The CLIME uses cross-validation to choose a common penalty level for all p columns. When p is large, it is computationally difficult. In fact, this prevented us from completing the simulation experiment for the CLIME for the larger $p \in \{300, 1000\}$.

7. Discussion

Since the scaled Lasso choose penalty levels adaptively in the estimation of each column of the precision matrix, it is expected to outperform methods using a fixed penalty level for all columns in the presence of heterogeneity of the diagonal of the precision matrix. Let $\widetilde{\Theta}(\lambda)$ be an estimator with columns

$$\widetilde{\Theta}_{*j}(\lambda) = \arg\min_{v \in \mathbb{R}^p} \Big\{ \|v\|_1 : \big\|\overline{\Sigma} v - e_j\big\|_\infty \le \lambda \Big\},\ j = 1, \ldots, p. \tag{46}$$

The CLIME is a symmetrization of this estimator $\widetilde{\Theta}(\lambda)$ with fixed penalty level for all columns. In the following example, the scaled Lasso estimator has a faster convergence rate than (46). The example also demonstrates the possibility of achieving the rate $d\lambda_0$ in Theorem 2 with unbounded $\|\Theta^*\|_2 \ge d^2$, when Theorem 1 is not applicable.

Example 1 *Let $p > n^2 + 3 + m$ with $(m, m^4(\log p)/n) \to (\infty, 0)$ and $4m^2 \le \log p$. Let $L_n(t) \approx \sqrt{(2/n)\log(1/t)}$ be as in (36). Let $\{J_1, J_2, J_3\}$ be a partition of $\{1, \ldots, p\}$ with $J_1 = \{1, 2\}$ and $J_2 = \{3, \ldots, 3+m\}$. Let $\rho_1 = \sqrt{1 - 1/m^2}$, $v = (v_1, \ldots, v_m)^T \in \mathbb{R}^m$ with $v_j^2 = 1/m$, $\rho_2 = c_0 m^{3/2} L_n(m/p) = o(1)$, and*

$$\Sigma^* = \begin{pmatrix} \Sigma^*_{J_1,J_1} & 0 & 0 \\ 0 & \Sigma^*_{J_2,J_2} & 0 \\ 0 & 0 & I_{p-m-3} \end{pmatrix},\ \Sigma^*_{J_1,J_1} = \begin{pmatrix} 1 & \rho_1 \\ \rho_1 & 1 \end{pmatrix},\ \Sigma^*_{J_2,J_2} = \begin{pmatrix} 1 & \rho_2 v^T \\ \rho_2 v & I_m \end{pmatrix}.$$

Model 1

	Spectrum norm				Matrix ℓ_1 norm				Frobenius norm			
p	SLasso	SLasso/LSE	GLasso	CLIME	SLasso	SLasso/LSE	GLasso	CLIME	SLasso	SLasso/LSE	GLasso	CLIME
30	2.46(0.07)	1.77(0.15)	2.49(0.14)	2.29(0.21)	2.95(0.10)	2.73(0.21)	3.09(0.11)	2.92(0.17)	4.20(0.11)	3.37(0.14)	4.24(0.26)	3.80(0.36)
60	2.68(0.05)	2.04(0.11)	2.94(0.05)	2.68(0.10)	3.12(0.08)	3.17(0.25)	3.55(0.07)	3.27(0.09)	6.41(0.09)	5.35(0.15)	7.15(0.15)	6.32(0.28)
90	2.75(0.04)	2.09(0.08)	3.07(0.03)	2.87(0.09)	3.21(0.07)	3.49(0.31)	3.72(0.06)	3.42(0.07)	8.09(0.10)	6.87(0.15)	9.25(0.12)	8.42(0.31)
150	2.84(0.03)	2.18(0.06)	3.19(0.02)	3.05(0.04)	3.29(0.07)	3.81(0.31)	3.88(0.06)	3.55(0.06)	10.79(0.11)	9.32(0.14)	12.55(0.09)	11.68(0.20)
300	2.93(0.02)	2.25(0.05)	3.29(0.01)	NA	3.39(0.05)	4.36(0.38)	4.06(0.05)	NA	15.83(0.09)	13.89(0.16)	18.44(0.09)	NA
1000	3.08(0.02)	2.38(0.08)	3.39(0.00)	NA	3.51(0.03)	5.13(0.37)	4.44(0.07)	NA	30.55(0.09)	26.68(0.19)	35.11(0.06)	NA

Model 2

	Spectrum norm				Matrix ℓ_1 norm				Frobenius norm			
p	SLasso	SLasso/LSE	GLasso	CLIME	SLasso	SLasso/LSE	GLasso	CLIME	SLasso	SLasso/LSE	GLasso	CLIME
30	0.72(0.08)	1.08(0.17)	0.82(0.07)	0.81(0.09)	1.27(0.15)	1.86(0.33)	1.49(0.15)	1.45(0.18)	1.86(0.10)	2.40(0.24)	1.84(0.09)	1.87(0.11)
60	1.06(0.05)	1.41(0.19)	1.15(0.06)	1.19(0.08)	1.93(0.15)	2.68(0.35)	2.21(0.12)	2.20(0.23)	3.27(0.08)	4.19(0.26)	3.18(0.13)	3.42(0.09)
90	1.48(0.04)	1.73(0.23)	1.54(0.05)	1.61(0.04)	2.58(0.15)	3.81(0.46)	2.89(0.16)	2.90(0.17)	4.42(0.07)	6.18(0.29)	4.40(0.11)	4.65(0.08)
150	1.96(0.03)	2.04(0.28)	2.02(0.05)	2.06(0.03)	3.25(0.17)	5.21(0.63)	3.60(0.15)	3.65(0.19)	5.95(0.06)	9.17(0.33)	6.19(0.16)	6.33(0.08)
300	2.88(0.02)	2.34(0.19)	2.89(0.02)	NA	4.45(0.13)	6.75(0.52)	4.92(0.17)	NA	9.26(0.05)	13.99(0.37)	9.79(0.05)	NA
1000	5.46(0.01)	4.88(0.03)	5.52(0.01)	NA	7.09(0.09)	10.26(0.53)	7.98(0.15)	NA	18.85(0.06)	26.08(0.30)	20.81(0.02)	NA

Model 3

	Spectrum norm				Matrix ℓ_1 norm				Frobenius norm			
p	SLasso	SLasso/LSE	GLasso	CLIME	SLasso	SLasso/LSE	GLasso	CLIME	SLasso	SLasso/LSE	GLasso	CLIME
30	1.84(0.09)	1.28(0.15)	2.08(0.10)	1.63(0.19)	2.30(0.12)	2.00(0.19)	2.59(0.10)	2.17(0.20)	2.69(0.09)	2.15(0.13)	2.91(0.16)	2.37(0.25)
60	2.18(0.07)	1.58(0.13)	2.63(0.04)	2.10(0.10)	2.63(0.10)	2.46(0.18)	3.10(0.05)	2.65(0.14)	4.10(0.08)	3.41(0.10)	4.84(0.08)	3.98(0.13)
90	2.34(0.06)	1.71(0.11)	2.84(0.03)	2.38(0.18)	2.77(0.10)	2.73(0.20)	3.30(0.06)	2.91(0.12)	5.19(0.08)	4.40(0.10)	6.25(0.08)	5.37(0.37)
150	2.51(0.05)	1.84(0.09)	3.06(0.02)	2.76(0.05)	2.93(0.09)	3.04(0.28)	3.45(0.04)	3.18(0.09)	6.93(0.08)	5.96(0.10)	8.43(0.07)	7.75(0.08)
300	2.70(0.05)	1.99(0.08)	3.26(0.01)	NA	3.10(0.07)	3.39(0.27)	3.58(0.03)	NA	10.18(0.08)	8.89(0.10)	12.41(0.04)	NA
1000	2.94(0.03)	2.16(0.07)	3.47(0.01)	NA	3.32(0.06)	4.07(0.32)	3.73(0.03)	NA	19.63(0.07)	17.04(0.15)	23.55(0.02)	NA

Table 1: Estimation errors under various matrix norms of scaled Lasso, GLasso and CLIME for three models.

The eigenvalues of $\Sigma^*_{J_1,J_1}$ *are* $1\pm\rho_1$*, those of* $\Sigma^*_{J_2,J_2}$ *are* $1\pm\rho_2, 1, \ldots, 1$*, and*

$$(\Sigma^*_{J_1,J_1})^{-1} = m^2\begin{pmatrix} 1 & -\rho_1 \\ -\rho_1 & 1 \end{pmatrix},\ (\Sigma^*_{J_2,J_2})^{-1} = \frac{1}{1-\rho_2^2}\begin{pmatrix} 1 & -\rho_2 v^T \\ -\rho_2 v & (1-\rho_2^2)I_m + \rho_2^2 vv^T \end{pmatrix}.$$

We note that $\mathrm{diag}(\Sigma^*) = I_p$*,* $d = m+1$ *is the maximum degree,* $\|\Theta^*\|_2 = 1/(1-\rho_1) \approx 2d^2$*, and* $\|\Theta^*\|_1 \approx 2d^2$*. The following statements are proved in the Appendix.*
(i) Let $\widehat{\Theta}$ *be the scaled Lasso estimator of* $\Theta^* = (\Sigma^*)^{-1}$ *with penalty level* $\lambda_0 = A\sqrt{(4/n)\log p}$*,* $A > 1$*, as in Theorem 2. Then, there exists a constant* M_1^* *such that*

$$P\Big\{\|\widehat{\Theta} - \Theta^*\|_2 \le \|\widehat{\Theta} - \Theta^*\|_1 \le M_1^* m L_n(m/p)\Big\} \to 1.$$

(ii) If $\rho_2 = c_0 m^{3/2} L_n(m/p)$ *with a sufficiently small constant* $c_0 > 0$*, then*

$$P\Big\{\inf_{\lambda>0}\|\widetilde{\Theta}(\lambda) - \Theta^*\|_2 \ge c_0 m^{3/2} L_n(m/p)/\sqrt{1+1/m}\Big\} \to 1.$$

Thus, the order of the ℓ_1 *and spectrum norms of the error of (46) for the best data dependent penalty level* λ *is larger than that of the scaled Lasso by a factor* $\sqrt{m}$*.*

Acknowledgments

This research is partially supported by National Science Foundation Grants DMS-11-06753 and DMS-12-09014.

Appendix A.

We provide all proofs in this appendix. We first prove the results in Section 4 since they are used to prove the results in Section 3.

A.1 Proof of Proposition 5

Lemma 20 in Ye and Zhang (2010) gives $\|v\|_q^q \le \|v\|_{(q,m)}^q + (a_q/m)^{q-1}\|v\|_1^q$. The rest of part (i) follows directly from definition. Lemma 20 in Ye and Zhang (2010) also gives $\|v\|_{(q,m)}^* \le \|v\|_{(q',m/a_q)} + m^{-1/q}\|v\|_1$. The rest of part (ii) is dual to the corresponding parts of part (i). Since $\|\overline{\Sigma}v\|_{(2,m)} = \max_{\|u\|_0=m,\|u\|_2=1} u^T\overline{\Sigma}v$ and $\|u^T\overline{\Sigma}^q\|_2 \le \kappa_+^q(m;\overline{\Sigma})$ for $q \in \{1/2,1\}$, part (iii) follows. □

A.2 Proof of Theorem 6

By the Karush-Kuhn-Tucker conditions,

$$(X^TX/n)h = z - \lambda g,\ \mathrm{sgn}(\widehat{\beta}_j)g_j \in \{0,1\},\ \|g\|_\infty \le 1. \tag{47}$$

Since $\zeta_{(2,m)}(z_B,\lambda_*)$ is the $\|\cdot\|_{(2,m)}$ norm of $(|z_B| - \lambda_*)_+$, (21) implies

$$|h^Tz| \quad\le\quad \lambda_*\|h\|_1 + \sum_{j\in B}|h_j|(|z_j| - \lambda_*)_+$$

$$\leq \quad \lambda_*\|h\|_1 + A_1\lambda_* m^{1/2}\|h_B\|^*_{(2,m)}. \tag{48}$$

Since $-h_j \mathrm{sgn}(\widehat{\beta}_j) \leq |\beta_j| - |\widehat{\beta}_j| \leq \min(|h_j|, -|h_j| + 2|\beta_j|)\ \forall\ j \in S^c$, (19) and (47) yield

$$\begin{aligned} -\lambda h^T g &\leq \lambda\|h_S\|_1 - \lambda\|h_{S^c}\|_1 + 2\lambda\|\beta_{S^c}\|_1 \\ &\leq \lambda\|h_S\|_1 - \lambda\|h_{S^c}\|_1 + 2\lambda\lambda_*(s_* - |S|). \end{aligned}$$

By applying the above bounds to the inner product of h and (47), we find

$$\begin{aligned} \|Xh\|_2^2/n &\leq A_1\lambda_* m^{1/2}\|h_B\|^*_{(2,m)} + (\lambda_* - \lambda)\|h_{S^c}\|_1 \\ &\quad +(\lambda + \lambda_*)\|h_S\|_1 + 2\lambda\lambda_*(s_* - |S|). \end{aligned}$$

Let $u = (A/\lambda)h$. It follows that when $A\lambda_* \leq \lambda$,

$$\frac{\|Xu\|_2^2}{n} \leq A_1 m^{1/2}\|u_B\|^*_{(2,m)} - (A-1)\|u_{S^c}\|_1 + (A+1)\|u_S\|_1 + 2A(s_* - |S|).$$

Since $X^T X/n = \overline{\Sigma}$, $u \in \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1)$ with $m_1 = s_* - |S|$. Since $h = \lambda u/A$ and $s_* = m_1 + |S|$, the conclusion follows from (23) and (24). □

A.3 Proof of Theorem 8

It follows from the scale equivariance of (26) that

$$\{\widehat{\beta}/\sigma^*, \widehat{\sigma}/\sigma^*\} = \{\widehat{b}, \widehat{\phi}\} = \arg\min_{b,\phi}\Big\{\|y^* - Xb\|_2^2/(2n\phi) + \lambda_0\|b\|_1 + \phi/2\Big\}, \tag{49}$$

where $y^* = y/\sigma^* = Xb^* + \varepsilon^*$ with $b^* = \beta/\sigma^*$ and $\varepsilon^* = \varepsilon/\sigma^*$. Our objective is to bound $\|X(\widehat{b} - b^*)\|_2^2/n$ and $\|\widehat{b} - b^*\|_q$ from the above and $\widehat{\sigma}/\sigma^*$ from both sides. To this end, we apply Theorem 6 to the Lasso estimator

$$\widehat{b}(\lambda) = \arg\min_b\Big\{\|y^* - Xb\|_2^2/(2n) + \lambda\|b\|_1\Big\}.$$

Let $z^* = z/\sigma^*$ and $h^*(\lambda) = \widehat{b}(\lambda) - b^*$. Since $\|y^* - Xb^*\|_2^2/n = \|\varepsilon^*\|_2^2/n = 1$,

$$\begin{aligned} 1 - \|y^* - X\widehat{b}(\lambda)\|_2^2/n &= h^*(\lambda)^T X^T(y^* - X\widehat{b}(\lambda))/n + h^*(\lambda)^T z^* \\ &= 2h^*(\lambda)^T z^* - \|Xh^*(\lambda)\|_2^2/n. \end{aligned}$$

Consider $\lambda \geq A\lambda_{*,0}$. Since (21) holds with $\{z, \lambda_*\}$ replaced by $\{z^*, \lambda_{*,0}\}$, we find as in the proof of Theorem 6 that

$$u(\lambda) \quad = \quad h^*(\lambda)A/\lambda \in \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1).$$

In particular, (48) gives

$$\begin{aligned} |h^*(\lambda)^T z^*| &\leq \lambda_{*,0}\|h^*(\lambda)\|_1 + A_1\lambda_{*,0} m^{1/2}\|h^*_B(\lambda)\|^*_{(2,m)} \\ &\leq (\lambda^2/A^2)\Big\{\|u(\lambda)\|_1 + A_1 m^{1/2}\|u_B(\lambda)\|^*_{(2,m)}\Big\}. \end{aligned}$$

Thus, the definition of M_σ^* in (29) gives

$$|2h^*(\lambda)^T z^* - \|Xh^*(\lambda)\|_2^2/n| < M_\sigma^* s_* \lambda^2.$$

We summarize the calculation in this paragraph with the following statement:

$$\lambda \geq A\lambda_{*,0} \;\Rightarrow\; \left|1 - \|y - X\widehat{b}(\lambda)\|_2^2/n\right| < M_\sigma^* s_* \lambda^2. \tag{50}$$

As in Sun and Zhang (2012), the convexity of the joint loss function in (49) implies

$$(\phi - \widehat{\phi})(\phi^2 - \|y - X\widehat{b}(\phi\lambda_0)\|_2^2/n) \geq 0,$$

so that $\widehat{\phi}$ can be bounded by testing the sign of $\phi^2 - \|y - X\widehat{b}(\phi\lambda_0)\|_2^2/n$. For $(\phi,\lambda) = (\phi_1, \phi_1\lambda_0)$, we have

$$\lambda^2 = \frac{\lambda_0^2}{1 + \lambda_0^2 M_\sigma^* s_*} \geq \frac{A^2\lambda_{*,0}^2}{1 - \eta_* + A^2\lambda_{*,0}^2 M_\sigma^* s_*} = A^2\lambda_{*,0}^2,$$

which implies $\|y - X\widehat{b}(\phi_1\lambda_0)\|_2^2/n > 1 - \phi_1^2\eta_0 = \phi_1^2$ by (50) and the definition of ϕ_1. This yields $\widehat{\phi} > \phi_1$. Similarly, $\widehat{\phi} < \phi_2$. The error bounds for the prediction and the estimation $\widehat{\beta}$ follow from Theorem 6 due to $A\lambda_{*,0} \leq \phi_1\lambda_0 < \widehat{\phi}\lambda_0 < \phi_2\lambda_0$. □

A.4 Proof of Lemma 9

By Proposition 5, $m^{1/2}\|u_B\|_{(2,m)}^* \leq \|u_B\|_1 + m^{1/2}\|u_B\|_{(2,4m)}$, so that for $u \in \mathscr{U}(\overline{\Sigma}, S, B; A, A_1, m, m_1)$,

$$u^T\overline{\Sigma}u + (A - A_1 - 1)\|u\|_1 \leq 2A\|u_S\|_1 + A_1 m^{1/2}\|u_B\|_2 + 2Am_1.$$

Let $\xi = A/(A - A_1 - 1)$ and $\xi_1 = A_1/(A - A_1 - 1)$. It follows that

$$\begin{aligned}
& (\xi/A)u^T\overline{\Sigma}u + \|u\|_1 \\
\leq\; & 2\xi\|u_S\|_1 + \xi_1 m^{1/2}\|u_B\|_2 + 2\xi m_1 \\
\leq\; & (2\xi|S| + \xi_1 m + 2\xi m_1)^{1/2}\{(2\xi + \xi_1)\|u_{S\cup B}\|_2^2 + 2\xi m_1\}^{1/2} \\
\leq\; & \{(2\xi s_* + \xi_1 m)/c_*\}^{1/2}\{(2\xi + \xi_1)u^T\Sigma u + 2\xi c_* m_1\}^{1/2} \\
\leq\; & \{(s_* \vee m)/c_*\}^{1/2}(2\xi + \xi_1)(u^T\Sigma u + c_* m_1)^{1/2}
\end{aligned} \tag{51}$$

due to $s_* = |S| + m_1$ and $c_*\|u_{S\cup B}\|_2^2 \leq u^T\Sigma u$. In terms of $\{\xi, \xi_1\}$, the condition of the Lemma can be stated as $\lambda^*\{(s_* \vee m)/c_*\}(2\xi + \xi_1)^2 \leq 1/2$. Thus,

$$u^T\Sigma u - u^T\overline{\Sigma}u \leq \lambda^*\|u\|_1^2 \leq u^T\Sigma u/2 + c_* m_1/2. \tag{52}$$

Inserting this inequality back into (51), we find that

$$(\xi/A)u^T\overline{\Sigma}u + \|u\|_1 \leq \{(s_* \vee m)/c_*\}^{1/2}(2\xi + \xi_1)(2u^T\overline{\Sigma}u + 2c_* m_1)^{1/2}.$$

If $(\xi/A)u^T\overline{\Sigma}u + \|u\|_1 \geq (\xi/A)(2u^T\overline{\Sigma}u + 2c_* m_1)/4$, we have

$$(\xi/A)u^T\overline{\Sigma}u + \|u\|_1 \leq \{(s_* \vee m)/c_*\}(2\xi + \xi_1)^2(4A/\xi).$$

Otherwise, we have $(\xi/A)u^T\overline{\Sigma}u+2\|u\|_1 \le (\xi/A)c_*m_1$. Consequently,

$$(\xi/A)u^T\overline{\Sigma}u+\|u\|_1 \le \max\Big\{\{(s_*\vee m)/c_*\}(2\xi+\xi_1)^2(4A/\xi), (\xi/A)c_*(s_*-|S|)\Big\}.$$

This and the definition of $\{M^*_{pred}, M^*_1\}$ yield (32) via

$$\xi M^*_{pred}+M^*_1 \le \max\Big\{\big(1\vee(m/s_*)\big)(2\xi+\xi_1)^2\frac{4}{\xi c_*}, \xi c_*(1-|S|/s_*)/A^2\Big\}.$$

Moreover, (52) gives $c_*\|u_{S\cup B}\|_2^2 \le u^T\Sigma u \le 2u^T\overline{\Sigma}u+2c_*m_1$, so that M^*_σ can be bounded via

$$\begin{aligned} & u^T\overline{\Sigma}u/(s_*A^2)+2\big(\|u\|_1+A_1m^{1/2}\|u_B\|^*_{(2,m)}\big)/(s_*A^2) \\ \le\ & M^*_{pred}+2(1+A_1)\|u\|_1/(s_*A^2)+(A_1/A)\Big\{m/s_*+\|u_B\|_2^2/(s_*A^2)\Big\} \\ \le\ & M^*_{pred}+2(1+A_1)M^*_1/A+(A_1/A)\Big(m/s_*+(2/c_*)M^*_{pred}+2(1-|S|/s_*)/A^2\Big). \end{aligned}$$

This gives (33). If in addition $B=\{1,\ldots,p\}$, then it yields (34)

$$M^*_2=\sup_{u\in\mathscr{U}}\|u\|_2^2/(s_*A^2) \le (2/c_*)M^*_{pred}+2(1-|S|/s_*)/A^2.$$

This completes the proof. □

The tail probability bound for $\zeta^*(z^*,\lambda_*,m)/\sigma^*$ in part (ii) of Proposition 10 uses the following version of the Lévy concentration inequality in the sphere.

Lemma 17 *Let* $\widetilde{\varepsilon}_m=\sqrt{2/(m-1/2)}\Gamma(m/2+1/2)/\Gamma(m/2)-1$, $U=(U_1,\ldots,U_{m+1})^T$ *be a uniform random vector in* $S^m=\{u\in\mathbb{R}^{m+1}:\|u\|_2=1\}$, $f(u)$ *a unit Lipschitz function in* S^m, *and* m_f *the median of* $f(U)$. *Then,*

$$P\{U_1>x\} \le (1+\widetilde{\varepsilon}_m)P\big\{N(0,1/(m-1/2)) > \sqrt{-\log(1-x^2)}\big\}, \tag{53}$$

$1<1+\widetilde{\varepsilon}_m<\exp\big(1/(4m-2)^2\big)$, *and*

$$\begin{aligned} P\{f(U)>m_f+x\} &\le P\Big\{U_1>x\sqrt{1-(x/2)^2}\Big\} \\ &\le (1+\widetilde{\varepsilon}_m)P\{N(0,1/(m-1/2))>x\}. \end{aligned} \tag{54}$$

PROOF. Since U_1^2 follows the beta$(1/2,m/2)$ distribution,

$$P\big\{U_1>x\big\}=\frac{\Gamma(m/2+1/2)/2}{\Gamma(m/2)\Gamma(1/2)}\int_{x^2}^1 t^{-1/2}(1-t)^{m/2-1}dt.$$

Let $y=\sqrt{-(m-1/2)\log(1-t)}$. We observe that $-t^{-1}\log(1-t)\le(1-t)^{-1/2}$ by inspecting the infinite series expansions of the two functions. This gives

$$\frac{e^{-y^2/2}dy}{t^{-1/2}(1-t)^{m/2-1}dt}=\frac{t^{1/2}e^{-y^2/2}(m-1/2)^{1/2}}{2(-\log(1-t))^{1/2}(1-t)^{m/2}} \ge 2^{-1}\sqrt{m-1/2}.$$

Since $y=\sqrt{-(m-1/2)\log(1-x^2)}$ when $t=x^2$, it follows that

$$P\{U_1>x\}\le(1+\widetilde{\varepsilon}_m)\int_{\sqrt{-(m-1/2)\log(1-x^2)}}^{\infty}(2\pi)^{-1/2}e^{-y^2/2}dt.$$

Let $\widetilde{A}=\{u\in S^m: f(u)\le m_f\}$, $H=\{u:u_1\le 0\}$, and $A_x=\cup_{v\in A}\{u\in S^m:\|u-v\|_2\le x\}$ for all $A\subset S^m$. Since $u\in\widetilde{A}_x$ implies $f(u)\le m_f+x$ and $P\{U\in\widetilde{A}\}\ge P\{U\in H\}$, the Lévy concentration inequality gives

$$P\{f(U)>m_f+x\}\le P\{U\notin H_x\}=P\Big\{U_1>x\sqrt{1-(x/2)^2}\Big\}.$$

The second inequality of (54) then follows from $(d/dx)\{-\log\{1-(x^2-x^4/4)\}-x^2\}\ge 0$ for $x^2\le 2$ and $\|U_1\|_\infty\le 1$.

It remains to bound $1+\widetilde{\varepsilon}_m$. Let $x=m+1/2$. Since

$$\frac{1+\widetilde{\varepsilon}_m}{1+\widetilde{\varepsilon}_{m+2}}=\frac{(m/2)\sqrt{m+3/2}}{(m/2+1/2)\sqrt{m-1/2}}=\frac{(x-1/2)\sqrt{x+1}}{(x+1/2)\sqrt{x-1}},$$

the infinite series expansion of its logarithm is bounded by

$$\log\Big(\frac{1+\widetilde{\varepsilon}_m}{1+\widetilde{\varepsilon}_{m+2}}\Big)=\frac{1}{2}\log\Big(\frac{1+1/x}{1-1/x}\Big)+\log\Big(\frac{1-1/(2x)}{1+1/(2x)}\Big)\le\frac{x^{-3}}{4}+\frac{x^{-5}}{5}+\cdots$$

Since $\{(x-1)^{-2}-(x+1)^{-2}\}/2=2x^{-3}+4x^{-5}+\cdots$ by Newton's binomial formula,

$$\log\Big(\frac{1+\widetilde{\varepsilon}_m}{1+\widetilde{\varepsilon}_{m+2}}\Big)\le\{(x-1)^{-2}-(x+1)^{-2}\}/16.$$

This gives $\log(1+\widetilde{\varepsilon}_m)\le 1/\{16(x-1)^2\}$. □

A.5 Proof of Proposition 10

(i) Let $L=L_1(k/p)$. Since $P\{N(0,\sigma^2/n)>\lambda_*\}=k/p$, $\lambda_*=\sigma L/\sqrt{n}$. Since $z_j=x_j^T\varepsilon/n\sim N(0,\sigma^2/n)$, $P\{\zeta_{(2,p)}(z,\lambda_*)>0\}\le 2k$ and

$$\begin{aligned}E\zeta_{(2,p)}^2(z,\lambda_*) &= p(\sigma^2/n)E(|N(0,1)|-L)_+^2\\ &= 2p(\sigma^2/n)\int_L^\infty(x-L)^2\varphi(x)dx.\end{aligned}$$

Let $J_k(t)=\int_0^\infty x^k e^{-x-x^2/(2t^2)}dx$. By definition

$$\frac{t^2\int_t^\infty(x-t)^2\varphi(x)dx}{\Phi(-t)}=\frac{t^2\int_0^\infty x^2e^{-tx-x^2/2}dx}{\int_0^\infty e^{-tx-x^2/2}dx}=\frac{\int_0^\infty u^2e^{-u-u^2/(2t^2)}du}{\int_0^\infty e^{-u-u^2/(2t^2)}du}=\frac{J_2(t)}{J_0(t)}.$$

Since $J_{k+1}+J_{k+2}/t^2=-\int_0^\infty x^{k+1}de^{-x-x^2/(2t^2)}=(k+1)J_k(t)$, we find

$$\frac{J_2(t)}{J_0(t)}=\frac{J_2(t)}{\{J_2(t)+J_3(t)/t^2\}/2+J_2(t)/t^2}\le\frac{1}{1/2+1/t^2}.$$

Thus, $E\zeta^2_{(2,p)}(z,\lambda_*) = 2p(\sigma^2/n)(k/p)L^{-2}J_2(L)/J_0(L) \le 2k\lambda_*^2 L^{-4}/(1/2+1/L^2)$.

Since $z_j = x_j^T\varepsilon/n$, $\left(\sum_{j\in B}(|z_j|-\lambda_*)_+^2\right)^{1/2}$ is a function of ε with the Lipschitz norm $\|X_B/n\|_2$. Thus, $\zeta_{(2,m)}(z,\lambda_*)$ is a function of ε with the Lipschitz norm $\max_{|B|=m}\|X_B/n\|_2 = \sqrt{\kappa_+(m)/n}$. In addition, since $\zeta_{(2,m)}(z,\lambda_*)$ is an increasing convex function of $(|z_j|-\lambda_*)_+$ and $(|z_j|-\lambda_*)_+$ are convex in ε, $\zeta_{(2,m)}(z,\lambda_*)$ is a convex function of ε. The mean of $\zeta_{(2,m)}(z,\lambda_*)$ is no smaller than its median. This gives (37) by the Gaussian concentration inequality (Borell, 1975).

(ii) The scaled version of the proof uses Lemma 17 with $m = n-1$ there. Let $U = \varepsilon/\|\varepsilon\|_2$, $z_j^* = x_j^T\varepsilon/(n\sigma^*) = (x_j/\sqrt{n})^T U$ and $z^* = X^T\varepsilon/(n\sigma^*)$. Since $z_j^* \sim U_1$, (53) yields the bound $P\{\zeta_{(2,p)}(z^*,\lambda_{*,0}) > 0\} \le (1+\varepsilon_n)2k$ and

$$E\zeta^2_{(2,p)}(z^*,\lambda_{*,0}) = pE(|U_1|-\lambda_*)_+^2 \le (1+\varepsilon_n)pE(|N(0,1)|-L)_+^2/(n-3/2).$$

The bound for $E\zeta^2_{(2,p)}(z^*,\lambda_{*,0})$ is then derived as in (i). Lemma 17 also gives

$$\begin{aligned} & P\big\{\pm(\zeta_{(2,m)}(z^*,\lambda_{*,0})-\mu_{(2,m)}) > x\sqrt{\kappa_+(m)}\big\} \\ \le\ & (1+\varepsilon_n)P\{|N(0,1/(n-3/2))| > x\} \end{aligned}$$

and

$$\begin{aligned} |E\zeta_{(2,m)}(z^*,\lambda_{*,0})-\mu_{(2,m)}| &\le (1+\varepsilon_n)\sqrt{\kappa_+(m)/(n-3/2)}E\big(N(0,1)\big)_+ \\ &= (1+\varepsilon_n)\sqrt{\kappa_+(m)/\{2\pi(n-3/2)\}}. \end{aligned}$$

The above two inequalities yield (38) and (39). □

A.6 Proof of Theorems 11, 12 and 13

The conclusions follow from Theorems 6 and 8 once (21) is proved to hold with the given probability. In Theorem 11, the tail probability bounds for $\zeta_{(0,p)}$ in Proposition 10 yield (21) with $A_1 = 0$. In Theorem 12, the moment bounds for $\zeta_{(0,p)}$ in Proposition 10 controls the excess noise in (21). In Theorem 13 (i), we need $A_1\lambda_* m^{1/2} \ge E\zeta_{(0,p)}(z,\lambda_*) + \sigma L_n(\varepsilon/p)\sqrt{\kappa_+(m)}$ by (37), so that the given lower bound of A_1 suffices due to $L_n(\varepsilon)/L_n(k/p) = L_1(\varepsilon/p)/L_1(k/p)$. The proof of Theorem 13 (ii) is nearly identical, with (38) and (39) in place of (38). We omit the details. □

A.7 Proof of Proposition 14

(i) By the ℓ_∞ constraint,

$$\|X^T(\varepsilon - X\widetilde{h}(\lambda))/n\|_{(2,k)} = \|X^T(y - X\widetilde{\beta}(\lambda))/n\|_{(2,k)} \le \lambda\sqrt{k}.$$

Thus, when $\lambda\sqrt{k} \le \|X^T\varepsilon/n\|_{(2,k)}/A$,

$$\|X^T\varepsilon/n\|_{(2,k)}(1-1/A) \le \|X^T\varepsilon/n\|_{(2,k)} - \lambda\sqrt{k} \le \|X^T X\widetilde{h}(\lambda)/n\|_{(2,k)}.$$

Thus, Proposition 5 (iii) gives (40).

(ii) Let $f(x) = (x - \widetilde{L}_1(k/p))_+ \wedge 1$ and $z^* = X^T\varepsilon/\|\varepsilon\|_2$. Since $z^* \sim N(0,\Sigma)$ and $\|f(z^*)\|_2$ has unit Lipschitz norm, the Gaussian concentration theorem gives

$$P\Big\{Ef(z^*) - f(z^*) \ge \sqrt{2\|\Sigma\|_2\log(1/\varepsilon)}\Big\} \le \varepsilon.$$

This implies $\mathrm{Var}(f(z^*)) \le 4\|\Sigma\|_2$. Since $Ef^2(z^*) \ge pP\{|N(0,1)| \ge L_1(k/p)\} = 2k$,

$$Ef(z^*) - \sqrt{2\|\Sigma\|_2 \log(1/\varepsilon)} \ge \sqrt{2k - 4\|\Sigma\|_2} - \sqrt{2\|\Sigma\|_2 \log(1/\varepsilon)} \ge \sqrt{k-1}.$$

This gives $P\{N_k \ge k\} \ge P\{f(z^*) > \sqrt{k-1}\} \ge 1-\varepsilon$ due to $N_k \ge f^2(z^*)$. Thus, (41) follows from $\|X^T\varepsilon/n\|_{(q,k)} \ge \sigma^* k^{1/q} \widetilde{L}_n(k/p)$ when $N_k \ge k$. The final conclusion follows from part (i) and large deviation for $(\sigma^*/\sigma)^2 \sim \chi^2_n/n$. □

Lemma 18 *Let $\chi^2_{m,j}$ be χ^2 distributed variables with m degrees of freedom. Then,*

$$E \max_{1\le j\le t} \chi^2_{m,j} \le \left(\sqrt{m} + \sqrt{2\log t}\right)^2, \quad t \ge 1.$$

PROOF. Let $f(t) = 2\log t - \int_0^\infty \min\left(1, tP\{N(0,1) > x\}\right) dx^2$. We first proof $f(t) \ge 0$ for $t \ge 2$. Let $L_1(x) = -\Phi^{-1}(x)$. We have $f(2) \ge 2\log 2 - 1 > 0$ and

$$f'(t) = 2/t - 2\int_{L_1(1/t)}^\infty P\big\{N(0,1) > x\big\} x dx \ge 2/t - 2\int_{L_1(1/t)}^\infty \varphi(x) dx = 0.$$

The conclusion follows from $P\{\chi_{m,j} > \sqrt{m} + x\} \le P\{N(0,1) > x\}$ for $x > 0$. □

A.8 Proof of Theorem 15

Let $h = \widehat{\beta} - \beta$ and $\widehat{\lambda} = \widehat{\sigma}\lambda_0$. Consider $J \subseteq \widehat{S} \setminus S$ with $m \le |J| \le m^*$. For any $j \in \widehat{S}$, it follows from the KKT conditions that $|x_j^T Xh/n| = |x_j^T(y - X\widehat{\beta} - \varepsilon)| \ge \widehat{\lambda} - |z_j|$. By the definition of $\kappa_+^*(m^*, S)$ and (25),

$$\begin{aligned}
\sum_{j\in J} (\widehat{\lambda} - |z_j|)_+^2 &\le \sum_{j\in J} |x_j^T Xh/n|^2 \\
&= (X_J^T Xh/n)^T (X_J^T Xh/n) \\
&\le \kappa_+^*(m^*, S)\|Xh\|_2^2/n \\
&\le \kappa_+^*(m^*, S) M_{pred}^* s_* \widehat{\lambda}^2. \qquad (55)
\end{aligned}$$

Since $\zeta_{(2,k)}(z_B^*, \lambda_{*,0})/k^{1/2} \downarrow k$ by Proposition 5 (i), the $\{z^*, \lambda_{0,*}\}$ version of (21) gives $\zeta_{(2,|J|)}(z^*, \lambda_{*,0})/|J|^{1/2} \le \zeta_{(2,m)}(z_B^*, \lambda_{*,0})/m^{1/2} \le \xi_1(A-1)\lambda_{*,0}$. Thus, with $z_j^* = z_j/\sigma^*$,

$$\begin{aligned}
\sum_{j\in J} (\widehat{\lambda} - |z_j|)_+^2 &\ge \sum_{j\in J} \left\{\widehat{\lambda} - \sigma^*\lambda_{*,0} - \sigma^*(|z_j^*| - \lambda_{*,0})_+\right\}_+^2 \\
&\ge \left\{|J|^{1/2}(\widehat{\lambda} - \sigma^*\lambda_{*,0}) - \sigma^*\zeta_{(2,|J|)}(z_B^*, \lambda_{*,0})\right\}_+^2 \\
&\ge |J|\left\{\widehat{\lambda} - \sigma^*\lambda_{*,0} - \sigma^*\xi_1(A-1)\lambda_{*,0}\right\}_+^2
\end{aligned}$$

Since $\lambda_{*,0}^2/(\lambda_0\phi_1)^2 = (\lambda_{*,0}/\lambda_0)^2(1+\eta_0) \le (1-\eta_*)/A^2 + \eta_*/A^2 = 1/A^2$, we have $\lambda_{*,0}\sigma^* < \lambda_{*,0}\widehat{\sigma}/\phi_1 \le \widehat{\lambda}/A$. The above inequalities and (55) yield

$$|J| \le \frac{\kappa_+^*(m^*, S) M_{pred}^* s_* \widehat{\lambda}^2}{\left\{\widehat{\lambda} - \sigma^*\lambda_{*,0} - \sigma^*\xi_1(A-1)\lambda_{*,0}\right\}_+^2} < \frac{\kappa_+^*(m^*, S) M_{pred}^* s_*}{\left\{1 - 1/A - \xi_1(1-1/A)\right\}_+^2} \le m^*.$$

Since $\widehat{S}\setminus S$ does not have a subset of size m^*, we have $|\widehat{S}\setminus S| < m^*$ as stated in (43). Let P_B be the projection to the linear span of $\{x_j, j \in B\}$. We have

$$\begin{aligned}
&\widehat{\sigma}^2 \ge \|P_{\widehat{S}}^{\perp} y\|_2^2/n = \overline{\sigma}^2 \ge \|P_{S\cup\widehat{S}}^{\perp} y\|_2^2/n = (\sigma^*)^2 - \|P_{S\cup\widehat{S}}\varepsilon\|_2^2/n,\\
&\|X\overline{h}\|_2^2/n = \|P_{\widehat{S}} y - X\beta\|_2^2/n = \|P_{\widehat{S}}\varepsilon\|_2^2/n + \|P_{\widehat{S}}^{\perp} X\beta\|_2^2/n. \qquad (56)
\end{aligned}$$

Let $N = \binom{p}{m^*}$. We have $\log N \le m^* \log(ep/m^*)$ by Stirling. By Lemma 18,

$$E\|P_{S\cup\widehat{S}}\varepsilon\|_2^2/\sigma^2 \le E\max_{|B|=m^*} \|P_{S\cup B}\varepsilon\|_2^2/\sigma^2 \le |S| + (\sqrt{m^*} + \sqrt{2\log N})^2.$$

Since $\max_{|B|=m}\|P_{S\cup B}\varepsilon\|_2$ is a unit Lipschitz function,

$$\begin{aligned}
\|P_{S\cup\widehat{S}}\varepsilon\|_2/\sigma &\le \big\{|S| + \big(\sqrt{m^*} + \sqrt{2m^*\log(ep/m^*)}\big)^2\big\}^{1/2} + L_1(\varepsilon/p)\\
&\le \sqrt{M_{lse}^* s_* \log(p/s_*)}
\end{aligned}$$

with probability ε/p. In addition, Theorem 8 gives $\|P_{\widehat{S}}^{\perp} X\beta\|_2^2 \le \|X\widehat{\beta} - X\beta\|_2^2 \le (1+\varepsilon_0)^2 M_{pred}^* s_* (\sigma\phi_2\lambda_0)^2$. Inserting these bounds into (56) yields (44). □

Lemma 19 *Suppose that the rows of $X \in \mathbb{R}^{n\times p}$ are iid $N(0,\Sigma)$ random vectors.*
(i) Let $Y = \mathrm{trace}(AX'X/n)$ and $\sigma^2 = \mathrm{trace}\{(A+A')\Sigma(A+A')\Sigma\}/2$ with a deterministic matrix A. Then, $EY = \mu = \mathrm{trace}(A\Sigma)$, $\mathrm{Var}(Y) = \sigma^2/n$ and

$$E\exp\Big\{t(Y-\mu)\Big\} \le \exp\Big\{-\frac{t\sigma}{\sqrt{2}} - \frac{n}{2}\log(1-\sqrt{2}t\sigma/n)\Big\}.$$

Consequently, for $0 < x \le 1$,

$$P\Big\{(Y-\mu)/\sigma > x\Big\} \le \exp\Big\{-\frac{n}{2}\Big(\sqrt{2}x - \log(1+\sqrt{2}x)\Big)\Big\} \le e^{-nx^2/4}.$$

(ii) Let R^* and $\overline{R}$ be the population and sample correlation matrices of X. Then,

$$P\Big\{|\overline{R}_{jk} - R_{jk}^*| > x\sqrt{1-(R_{jk}^*)^2}\Big\} \le 2P\big\{|t_n| > n^{1/2}x\big\}$$

where t_n has the t-distribution with n degrees of freedom. In particular, for $n \ge 4$,

$$P\Big\{|\overline{R}_{jk} - R_{jk}^*| > \sqrt{2}x\Big\} \le 2e^{1/(4n-2)^2} P\big\{|N(0,1/n)| > x\big\},\ 0 \le x \le 1.$$

PROOF. (i) This part can be proved by computing the moment generating function with $t\sigma/n = x/(1+\sqrt{2}x)$. We omit details. For $0 < x < 1$,

$$f(x) = \frac{\sqrt{2}x - \log(1+\sqrt{2}x)}{x^2} = \int_0^{\sqrt{2}x} \frac{udu}{x^2(1+u)} = \int_0^{\sqrt{2}} \frac{udu}{1+xu} \ge f(1) > 1/2.$$

(ii) Conditionally on $\overline{\Sigma}_{kk}$, $\overline{\Sigma}_{jk}/\overline{\Sigma}_{kk} \sim N(\Sigma_{jk}/\Sigma_{kk}, (1-(R_{jk}^*)^2)\Sigma_{jj}/(n\overline{\Sigma}_{kk}))$. Thus,

$$z_{jk} = \Big(\frac{n\overline{\Sigma}_{kk}}{(1-(R_{jk}^*)^2)\Sigma_{jj}}\Big)^{1/2}\Big(\frac{\overline{\Sigma}_{jk}}{\overline{\Sigma}_{kk}} - \frac{\Sigma_{jk}}{\Sigma_{kk}}\Big)$$

$$= \Big(\frac{n}{1-(R^*_{jk})^2}\Big)^{1/2}\Big(\overline{R}_{jk}\frac{\overline{\Sigma}^{1/2}_{jj}}{\Sigma^{1/2}_{jj}} - R_{jk}\frac{\overline{\Sigma}^{1/2}_{kk}}{\Sigma^{1/2}_{kk}}\Big)$$

is a $N(0,1)$ variable independent of $\overline{\Sigma}_{kk}$. Consequently,

$$\frac{n^{1/2}|\overline{R}_{jk}-R_{jk}|}{(1-(R^*_{jk})^2)^{1/2}} = \frac{|z_{jk}+z_{kj}|}{\overline{\Sigma}^{1/2}_{jj}/\Sigma^{1/2}_{jj}+\overline{\Sigma}^{1/2}_{kk}/\Sigma^{1/2}_{kk}} \le |t_{jk}|\vee|t_{kj}|$$

with $t_{jk} = z_{jk}\Sigma^{1/2}_{kk}/\overline{\Sigma}^{1/2}_{kk} \sim t_n$. Let U_1 be a uniformly distributed variable in the unit sphere of $\mathbb{R}^{n+1}$. Since $t_n^2/n \sim U_1^2/(1-U_1^2)$, Lemma 17 provides

$$P\big\{t_n^2/n > e^{x^2}-1\big\} = P\big\{U_1^2 > 1-e^{-x^2}\big\} \le 2e^{1/(4n-2)^2}P\Big\{N(0,1/(n-1/2)) > x\Big\}.$$

The conclusion follows from $e^{x^2n/(n-1/2)}-1 \le 2x^2$ for $0<x\le 1$. □

A.9 Proof of Proposition 4

Since $\widetilde{\Theta}_{jj} = 1/\widehat{\sigma}_j^2$ and $\max\{C_0\lambda_0, C_1 s_{*,j}\lambda_0^2\} \le 1/4$, (17) and the condition on σ^*_j implies

$$|\widetilde{\Theta}_{jj}/\Theta^*_{jj}| \le (5/4)^3 \le 2,\ |\widetilde{\Theta}_{jj}/\Theta^*_{jj}-1| \le \{(5/4)^2C_0+5/4+1\}\lambda_0.$$

It follows from (7), (17) and the condition on $\overline{D} = \mathrm{diag}(\overline{\Sigma}_{jj}, j\le p)$ that

$$\begin{aligned}
\big\|\widetilde{\Theta}_{*,j}-\Theta^*_{*,j}\big\|_1 &= \big\|-\widehat{\beta}_{*,j}\widetilde{\Theta}_{jj}-\Theta^*_{*,j}\big\|_1 \\
&\le \big\|(\widehat{\beta}_{-j,j}-\beta_{-j,j})\widetilde{\Theta}_{jj}\big\|_1 + \big\|\Theta^*_{*,j}(\widetilde{\Theta}_{jj}/\Theta^*_{jj}-1)\big\|_1 \\
&\le \big\|\widehat{D}_{-j}^{-1/2}\big\|_\infty\big\|\widehat{D}_{-j}^{1/2}(\widehat{\beta}_{-j,j}-\beta_{-j,j})\big\|_1\widetilde{\Theta}_{jj} + \big\|\Theta^*_{*,j}\big\|_1|\widetilde{\Theta}_{jj}/\Theta^*_{jj}-1| \\
&\le (5/2)\Theta^*_{jj}\big\|D_{-j}^{-1/2}\big\|_\infty(\Theta^*_{jj})^{-1/2}C_2 s_{*,j}\lambda_0 + \big\|\Theta_{*,j}\big\|_1\{(3/2)C_0+5/2\}\lambda_0 \\
&\le C\Big\{(\big\|D_{-j}^{-1}\big\|_\infty\Theta^*_{jj})^{1/2}s_{*,j}\lambda_0 + \big\|\Theta_{*,j}\big\|_1\lambda_0\Big\}
\end{aligned}$$

with $C = \max(5C_2/2, 3C_0/2+5/2)$. This gives (12) due to $\|\widehat{\Theta}-\Theta^*\|_1 \le 2\|\widetilde{\Theta}-\Theta^*\|_1$ by (9). Similarly,

$$\begin{aligned}
\big\|\widetilde{\Omega}_{*,j}-\Omega^*_{*,j}\big\|_1 &= \big\|-\widehat{D}_{-j}^{1/2}\widehat{\beta}_{*,j}\widetilde{\Theta}_{jj}\widehat{D}_{jj}^{1/2}-\Omega^*_{*,j}\big\|_1 \\
&\le \big\|\widehat{D}_{-j}^{1/2}(\widehat{\beta}_{-j,j}-\beta_{-j,j})\big\|_1\widetilde{\Theta}_{jj}\widehat{D}_{jj}^{1/2} \\
&\quad +\big\|\widehat{D}_{-j}^{1/2}D_{-j}^{-1/2}\Omega^*_{*,j}(\widetilde{\Theta}_{jj}/\Theta^*_{jj})(\widehat{D}_{jj}/D_j)^{1/2}-\Omega^*_{*,j}\big\|_1 \\
&\le C\big\{(\Theta^*_{jj})^{-1/2}s_{*,j}\lambda_0\Theta^*_{jj}D_{jj}^{1/2} + \big\|\Omega_{*,j}\big\|_1\lambda_0\big\}
\end{aligned}$$

This gives (13) due to $D_{jj}\Theta^*_{jj} = \Omega^*_{jj}$. We omit an explicit calculation of C.

Let $\chi^2_{n,j} = n\Theta^*_{jj}(\sigma^*_j)^2$. When $\chi^2_{n,j} \sim \chi^2_n$, we have

$$|\widetilde{\Theta}_{jj}/\Theta^*_{jj}-1| \le \{(5/4)^2+5/4\}C_1 s_{*,j}\lambda_0^2 + (4/3)|\chi^2_{n,j}/n-1|$$

It follows from Lemma 19 that $P\{|\chi^2_{n,j}/n-1|>\sqrt{2}x\}\le 2e^{-nx^2/4}$ for $x\le 1$. Let $a_j=\|\Theta_{*,j}\|_1$, $t=\max\{M\max_j a_j/\sqrt{n},\tau_n(\Theta^*)\}$ and $B_0=\{j: a_j\le\sqrt{8}t\}$. By definition $t\le M\tau_n(\Theta^*)$ and $nt^2/a_j^2\ge M^2$. It follows that

$$\begin{aligned} P\Big\{\max_j|\chi^2_{n,j}/n-1|a_j>4t\Big\} &\le |B_0|e^{-n/4}+\sum_{j\notin B_0}\exp(-2nt^2/a_j^2)\\ &\le pe^{-n/4}+e^{-M^2}\sum_{j\notin B_0}\exp\Big(-n\tau^2(\Theta^*)/a_j^2\Big).\\ &\le pe^{-n/4}+e^{-M^2}. \end{aligned}$$

Thus, $\max_j|\widetilde{\Theta}_{jj}/\Theta^*_{jj}-1|a_j=O_P\big(\tau_n(\Theta^*)+\max_j s_{*,j}a_j\lambda_0^2\big)$. □

A.10 Proof of Theorem 2

We need to verify conditions (17) and (18) in order to apply Proposition 4. Since $\Theta^*_{jj}(\sigma^*_j)^2\sim\overline{\Sigma}_{jj}/\Sigma^*_{jj}\sim\chi^2_n/n$, (18) follows from Lemma 19 (i) with $\lambda_0\asymp\sqrt{(\log p)/n}$. Moreover, the condition $P\{(1-\varepsilon_0)^2\le\chi^2_n/n\le(1+\varepsilon_0)^2\}\le\varepsilon/p$ holds with small ε_0 and ε since $\sqrt{(\log p)/n}=\lambda_0/(2A)$ is assumed to be sufficiently small. We take $\varepsilon_0=0$ in (10) since its value does not change the order of $s_{*,j}$.

If we treat $\overline{\Sigma}^{1/2}_{kk}\beta_k$ as the regression coefficient in (4) for the standardized design vector $\overline{\Sigma}^{-1/2}_{kk}x_k$, $k\ne j$, Theorem 11 (ii) asserts that the conclusions of Theorem 8 hold with probability $1-3\varepsilon/p$ for each j, with $\lambda_0=A\sqrt{4(\log p)/n}$, $A_1=0$ and $\varepsilon\asymp 1/\sqrt{\log p}$. By the union bound, the conclusions of Theorem 8 holds simultaneously for all j with probability $1-3\varepsilon$. Moreover, (17) is included in the conclusions of Theorem 8 when M^*_σ and M^*_1 are uniformly bounded in the p regression problems with large probability. Thus, it suffices to verify the uniform boundedness of these quantities.

We use Lemma 9 to verify the uniform boundedness of M^*_σ and M^*_1 with $A_1=0$, $B_j=S_j$, $m_j=0$ and $\{\overline{\Sigma},\Sigma^*\}$ replaced by $\{\overline{R}_{-j,-j},R^*_{-j,-j}\}$. Note that the Gram matrix for the regression problem in (4) is $\overline{R}_{-j,-j}$, which is random and dependent on j, so that M^*_σ and M^*_1 are random and dependent on j with the random design. It follows from Lemma 19 (ii) that

$$\max_{k\ne j}\|\overline{R}_{k,-j}-R^*_{k,-j}\|_\infty\le\max_{j,k}|\overline{R}_{k,j}-R_{k,j}|\le L_n(5\varepsilon/p^2)$$

with probability $1-\varepsilon$. We may take $L_n(5\varepsilon/p^2)=2\sqrt{(\log p)/n}$ with $\varepsilon\asymp 1/\sqrt{\log p}$. This yields the first condition of Lemma 9 with $\lambda^*=2\sqrt{(\log p)/n}\asymp\lambda_0$. The second condition $c_*\|u_S\|_2^2\le u^TR^*_{-j,-j}u$ follows from (11). The third condition translates to $\max_{j\le p}\lambda_0 s_{*,j}\le c_0$, which is imposed in Theorem 2. Thus, all conditions of Lemma 9 hold simultaneously for all j with large probability. The proof is complete since the conclusions of Lemma 9 with $m=m_j=0$ guarantee the uniform boundedness of M^*_σ and M^*_1. □

A.11 Proof of Theorem 3

The proof is parallel to that of Theorem 2. Since the smaller $\lambda_{*,0}=L_{n-3/2}(k/p)$ is used, we need to apply Theorem 13 (ii) with $A_1>0$, $m=m_j>0$ and typically much larger B_j than S_j. Since the condition $m_j\le C_0 s_{*,j}$ is impose in (15), the conclusions of Lemma 9 still guarantee the uniform boundedness of M^*_σ and M^*_1. The verification of the conditions of Lemma 9 is identical to the case of larger $\lambda_{*,0}$ in Theorem 2. The only difference is the need to verify that condition (15) uniformly guarantees

the condition on A_1 in Theorem 13 (ii), where κ_+/m has the interpretation of $\kappa_+(m_j;\overline{R}_{-j,-j})/m_j$, which depends on j and random $\overline{R}$. Anyway, it suffices to verify $\kappa_+(m_j;\overline{R}_{-j,-j})/m_j \le \psi_j$ simultaneously for all j with large probability.

We verify $\kappa_+(m_j;\overline{R}_{-j,-j})/m_j \le \psi_j$ with the same argument as in Lemma 9. For any vector u with $\|u\|_0 = m_j$ and $\|u\|_2 = 1$, it holds with probability $1-\varepsilon$ that

$$\left|u^T(\overline{R}_{-j,-j} - R_{-j,-j})u\right| \le \max_{j,k}|\overline{R}_{k,j} - R_{k,j}| \sum_{j,k}|u_j u_k| \le L_n(5\varepsilon/p^2)m_j.$$

Thus, it follows from the definition of $\kappa_+(m;\Sigma)$ in (16) that $\kappa_+(m_j;\overline{R}_{-j,-j})/m_j \le \kappa_+(m_j;R_{-j,-j})/m_j + L_n(5\varepsilon/p^2) = \psi_j$ for all j. This completes the proof. □

A.12 Proof of Example 1

(i) Let $s_{*,j} = d_j = \#\{k : \Theta^*_{jk} \ne 0\} \le m+1$. We have $\max_j(1+s_{*,j})\lambda_0 \le (m+2)\lambda_0 \to 0$. Let $B_j = \{k \ne j : \Theta^*_{kj} \ne 0\}$. Since $B_j = J_1 \setminus \{j\}$ for $j \in J_1$, (11) holds with

$$\inf\left\{u^T(R^*_{-j,-j})u/\|u_{B_j}\|_2^2 : u_{B_j} \ne 0\right\} \ge 1-\rho_2 \to 1.$$

Thus, Theorem 2 is directly applicable to this example.

Next, we calculate the error bound in (12) and (14). Since $d_j(\Theta^*_{jj})^{1/2} = 2/(1-\rho_1^2)^{1/2} = 2m$ for $j \in J_1$ and $d_j(\Theta^*_{jj})^{1/2} \le (m+1)/(1-\rho_2^2)^{1/2} \le 2m$ for $j \in J_2$,

$$(\|D_{-j}^{-1}\|_\infty \Theta^*_{jj})^{1/2} s_{*,j}\lambda_0 = (\Theta^*_{jj})^{1/2} s_{*,j}\lambda_0 \le 2m\lambda_0.$$

In addition, $\|\Theta_{*,j}\|_1 \le 2m^2$ for $j \in J_1$ and $\|\Theta_{*,j}\|_1 \le (1+\rho_2\|v\|_1)/(1-\rho_2^2) \le 3/2+o(1)$ for $j \in J_2$, so that for $t = \sqrt{(2/n)\log p}$,

$$\sum_j \exp(-nt^2/\|\Theta_{*,j}\|_1^2) \le 2\exp\Big(-\frac{2\log p}{4m^2}\Big) + p\exp\Big(-\frac{2\log p}{3/2+o(1)}\Big) \to 0.$$

It follows that the quantities in (14) are bounded by

$$\max_{j\le p} s_{*,j}\|\Theta^*_{*,j}\|_1\lambda_0^2 \le 2(m\lambda_0)^2,\ \tau_n(\Theta^*) \le \sqrt{(2/n)\log p} \le \lambda_0/(A\sqrt{2}).$$

Since $m\lambda_0 \to 0$, the error for the scaled Lasso is of the order $m\lambda_0$ by Theorem 2. The conclusion follows since $L_n(m/p) = (1+o(1))\sqrt{(2/n)\log p}$ when $4m^2 \le \log p$.

(ii)Let $\widetilde{\lambda} = \max_j \|\overline{\Sigma}_{*,j} - \Sigma^*_{*,j}\|_\infty$ and $\widetilde{\lambda}^* = \rho_2/\sqrt{m} + \widetilde{\lambda}$. Since $\mathrm{diag}(\Sigma^*) = I_n$,

$$\widetilde{\lambda} \lesssim L_n(1/p) \ll \rho_2/\sqrt{m}, \quad \widetilde{\lambda}^* = (1+o(1))\rho_2/\sqrt{m} = (1+o(1))c_0 m L_n(m/p).$$

For $\lambda \ge \widetilde{\lambda}^*$, e_3 is feasible for (46) with $j = 3 \in J_2$, so that $\|\widetilde{\Theta}_{*j}(\lambda)\|_1 \le 1$. Since $\|\Theta^*_{J_2,3}\|_1 \ge 1 + m^{1/2}\rho_2$,

$$m^{1/2}\rho_2 \le \inf_{\lambda\ge\widetilde{\lambda}^*} \|\widetilde{\Theta}_{J_2,3}(\lambda) - \Theta^*_{J_2,3}(\lambda)\|_1 \le (m+1)^{1/2} \inf_{\lambda\ge\widetilde{\lambda}^*} \|\widetilde{\Theta}_{J_2,3}(\lambda) - \Theta^*_{J_2,3}(\lambda)\|_2.$$

It follows that for $\lambda \geq \widetilde{\lambda}^*$, $\widetilde{\Theta}(\lambda)$ is suboptimal in the sense of

$$\inf_{\lambda \geq \widetilde{\lambda}^*} \|\widetilde{\Theta}(\lambda) - \Theta^*(\lambda)\|_2 \geq \sqrt{m/(1+m)}\rho_2 = c_0 m^{3/2} L_n(m/p)/\sqrt{1+1/m}.$$

Consider $\lambda \leq \widetilde{\lambda}^*$. Let $\beta_{-j,j} = -\Theta^*_{-j,j}/\Theta^*_{jj}$, $\widetilde{\beta}_{-j,j}(\lambda) = -\widetilde{\Theta}_{-j,j}(\lambda)/\widetilde{\Theta}_{jj}(\lambda)$, $\sigma_j = (\Theta^*_{j,j})^{-1/2}$, and $\widetilde{h}_j(\lambda) = \widetilde{\beta}_{-j,j}(\lambda) - \beta_{-j,j}$. By (46), $\|X^T_{-j}(x_j - X_{-j}\widetilde{\beta}(\lambda))/n\|_\infty \leq \lambda/\widetilde{\Theta}_{jj}(\lambda)$. Since $m(\log p)/n \to 0$ and $\|\Sigma^*\|_2 \leq 2$, $P\{\kappa_+(m;\overline{\Sigma}) \leq 3\} \to 1$. Thus, by Proposition 14, there exist positive constants $\{c_1, c_2\}$ such that

$$\min_j P\Big\{ \inf_{\lambda/\widetilde{\Theta}_{jj}(\lambda) \leq c_1 \sigma_j L_n(m/p)} \|\widetilde{h}_j(\lambda)\|_2 \geq c_2 \sigma_j \sqrt{m} L_n(m/p) \Big\} \to 1.$$

For $\widetilde{\Theta}_{jj}(\lambda) \geq \Theta^*_{jj}/2$,

$$\begin{aligned} \|\widetilde{h}_j(\lambda)\|_2 &= \|\widetilde{\Theta}_{-j,j}(\widetilde{\lambda})/\widetilde{\Theta}_{jj}(\lambda) - \Theta^*_{-j,j}/\Theta^*_{jj}\|_2 \\ &\leq \|\widetilde{\Theta}_{-j,j}(\widetilde{\lambda}) - \Theta^*_{-j,j}\|_2/\widetilde{\Theta}_{jj}(\lambda) + \|\beta_{-j,j}\|_2 |\widetilde{\Theta}_{jj}(\lambda) - \Theta^*_{jj}|/\widetilde{\Theta}_{jj}(\lambda) \\ &\leq \|\widetilde{\Theta}_{*,j}(\widetilde{\lambda}) - \Theta^*_{*,j}\|_2 (1 + \|\beta_{-j,j}\|_2)/(\Theta^*_{jj}/2). \end{aligned}$$

For $j = 1$, $\Theta^*_{jj} = m^2$ and $\|\beta_{-j,j}\|_2 = \rho_1$, so that $\|\widetilde{\Theta}_{*,j}(\widetilde{\lambda}) - \Theta^*_{*,j}\|_2 \geq m^2 \|\widetilde{h}_j(\lambda)\|_2/4$ when $\widetilde{\Theta}_{jj}(\lambda) \geq \Theta^*_{jj}/2$. Since $\|\widetilde{\Theta}_{*,j}(\widetilde{\lambda}) - \Theta^*_{*,j}\|_2 \geq m^2/2$ when $\widetilde{\Theta}_{jj}(\lambda) \leq \Theta^*_{jj}/2$,

$$\inf_{\lambda \leq \widetilde{\lambda}^*} \|\widetilde{\Theta}(\lambda) - \Theta^*\|_2 \geq \min\Big(m^2 \|\widetilde{h}_1(\lambda)\|_2/4, m^2/2\Big).$$

Pick $0 < c_0 < \min(c_1/2, c_2/4)$. Since $\sigma_1 = (\Theta^*_{11})^{-1/2} = 1/m$,

$$\begin{aligned} & P\Big\{ \inf_{\lambda \leq \widetilde{\lambda}^*} \|\widetilde{\Theta}(\lambda) - \Theta^*\|_2 \leq \min\big(m^2/2, (c_2/4) m^{3/2} L_n(m/p)\big) \Big\} \\ \leq\ & P\Big\{ \widetilde{\lambda}^* > (m^2/2)(c_1/m) L_n(m/p) \Big\} + o(1) = o(1). \end{aligned}$$

Since $L_n(m/p) \to 0$ implies $\min\big(m^2/2, (c_2/4) m^{3/2} L_n(m/p)\big) \geq c_0 m^{3/2} L_n(m/p)$, the conclusion follows. □

References

F. Abramovich and V. Grinshtein. Map model selection in gaussian regression. *Electr. J. Statist.*, 4: 932–949, 2010.

A. Antoniadis. Comments on: ℓ_1-penalization for mixture regression models. *Test*, 19(2):257–258, 2010.

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, 2008.

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

L. Birge and P. Massart. Gaussian model selection. *J. Eur. Math. Soc.*, 3:203–268, 2001.

L. Birge and P. Massart. Minimal penalties for gaussian model selection. *Probability Theory Related Fields*, 138:33–73, 2007.

C. Borell. The brunn-minkowski inequality in gaussian space. *Invent. Math.*, 30:207–216, 1975.

F. Bunea, A. Tsybakov, and M.H. Wegkamp. Aggregation for gaussian regression. *Ann. Statist.*, 35: 1674–1697, 2007.

T. Cai, L. Wang, and G. Xu. Shifting inequality and recovery of sparse signals. *IEEE Transactions on Signal Processing*, 58:1300–1308, 2010.

T. Cai, W. Liu, and X. Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106:594–607, 2011.

E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Trans. on Information Theory*, 51:4203–4215, 2005.

E. J. Candès and T. Tao. The dantzig selector: statistical estimation when p is much larger than n (with discussion). *Annals of Statistics*, 35:2313–2404, 2007.

D. L. Donoho and I. Johnstone. Minimax risk over ℓ_p–balls for ℓ_q–error. *Probability Theory and Related Fields*, 99:277–303, 1994.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical Lasso. *Biostatistics*, 9:432–441, 2008.

P. J. Huber and E. M. Ronchetti. *Robust Statistics*, pages 172–175. Wiley, second edition, 2009.

V. Koltchinskii. The dantzig selector and sparsity oracle inequalities. *Bernoulli*, 15:799–828, 2009.

C. Lam and J. Fan. Sparsistency and rates of convergence in large covariance matrices estimation. *Annals of Statistics*, 37:4254–4278, 2009.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the Lasso. *Annals of Statistics*, 34:1436–1462, 2006.

S. Negahban, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers. *Statistical Science*, 27: 538–557, 2012.

G. Raskutti, M. J. Wainwright, and B. Yu. Minimax rates of estimation for high–dimensional linear regression over ℓ_q–balls. *IEEE Trans. Info. Theory*, 57:6976–6994, 2011.

P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. Model selection in gaussian graphical models: High-dimensional consistency of ℓ_1-regularized MLE. *In Advances in Neural Information Processing Systems (NIPS)*, 21, 2008.

G. Rocha, P. Zhao, and B. Yu. A path following algorithm for sparse pseudo-likelihood inverse covariance estimation (splice). Technical report, University of California, Berkeley, 2008.

A.J. Rothman, P.J. Bickel, E. Levina, and J. Zhu. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

T. Sun and C.-H. Zhang. Scaled sparse linear regression. *Biometrika*, 99:879–898, 2012.

S. van de Geer. The deterministic Lasso. Technical Report 140, ETH Zurich, Switzerland, 2007.

S. van de Geer and P. Bühlmann. On the conditions used to prove oracle results for the Lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2009.

S. Yang and E. D. Kolaczyk. Target detection via network filtering. *IEEE Transactions on Information Theory*, 56(5):2502–2515, 2010.

F. Ye and C.-H. Zhang. Rate minimaxity of theLasso and Dantzig selector for the ℓ_q loss in ℓ_r balls. *Journal of Machine Learning Research*, 11:3481–3502, 2010.

M. Yuan. Sparse inverse covariance matrix estimation via linear programming. *Journal of Machine Learning Research*, 11:2261–2286, 2010.

M. Yuan and Y. Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics*, 38:894–942, 2010.

C.-H. Zhang and J. Huang. The sparsity and bias of the Lasso selection in high-dimensional linear regression. *Annals of Statistics*, 36(4):1567–1594, 2008.

C.-H. Zhang and T. Zhang. A general theory of concave regularization for high dimensional sparse estimation problems. *Statistical Science*, 27(4):576–593, 2012.

T. Zhang. Some sharp performance bounds for least squares regression with L_1 regularization. *Ann. Statist.*, 37(5A):2109–2144, 2009.

Journal of Machine Learning Research 14 (2013) 3419-3440 Submitted 8/12; Revised 7/13; Published 11/13

Consistent Selection of Tuning Parameters via Variable Selection Stability

Wei Sun SUN244@PURDUE.EDU
Department of Statistics
Purdue University
West Lafayette, IN 47907, USA

Junhui Wang JUNHUI@UIC.EDU
Department of Mathematics, Statistics, and Computer Science
University of Illinois at Chicago
Chicago, IL 60607, USA

Yixin Fang YIXIN.FANG@NYUMC.ORG
Departments of Population Health and Environmental Medicine
New York University
New York, NY 10016, USA

Editor: Xiaotong Shen

Abstract

Penalized regression models are popularly used in high-dimensional data analysis to conduct variable selection and model fitting simultaneously. Whereas success has been widely reported in literature, their performances largely depend on the tuning parameters that balance the trade-off between model fitting and model sparsity. Existing tuning criteria mainly follow the route of minimizing the estimated prediction error or maximizing the posterior model probability, such as cross validation, AIC and BIC. This article introduces a general tuning parameter selection criterion based on variable selection stability. The key idea is to select the tuning parameters so that the resultant penalized regression model is stable in variable selection. The asymptotic selection consistency is established for both fixed and diverging dimensions. Its effectiveness is also demonstrated in a variety of simulated examples as well as an application to the prostate cancer data.

Keywords: kappa coefficient, penalized regression, selection consistency, stability, tuning

1. Introduction

The rapid advance of technology has led to an increasing demand for modern statistical techniques to analyze data with complex structure such as the high-dimensional data. In high-dimensional data analysis, it is generally believed that only a small number of variables are truly informative while others are redundant. An underfitted model excludes truly informative variables and may lead to severe estimation bias in model fitting, whereas an overfitted model includes the redundant uninformative variables, increases the estimation variance and hinders the model interpretation. Therefore, identifying the truly informative variables is regarded as the primary goal of the high-dimensional data analysis as well as its many real applications such as health studies (Fan and Li, 2006).

Among other variable selection methods, penalized regression models have been popularly used, which penalize the model fitting with various regularization terms to encourage model sparsity, such as the lasso regression (Tibshirani, 1996), the smoothly clipped absolute deviation (SCAD, Fan and Li, 2001), the adaptive lasso (Zou, 2006), and the truncated l_1-norm regression (Shen et al., 2012). In the penalized regression models, tuning parameters are often employed to balance the trade-off between model fitting and model sparsity, which largely affects the numerical performance and the asymptotic behavior of the penalized regression models. For example, Zhao and Yu (2006) showed that, under the irrepresentable condition, the lasso regression is selection consistent when the tuning parameter converges to 0 at a rate slower than $O(n^{-1/2})$. Analogous results on the choice of tuning parameters have also been established for the SCAD, the adaptive lasso, and the truncated l_1-norm regression. Therefore, it is of crucial importance to select the appropriate tuning parameters so that the performance of the penalized regression models can be optimized.

In literature, many classical selection criteria have been applied to the penalized regression models, including cross validation (Stone, 1974), generalized cross validation (Craven and Wahba, 1979), Mallows' C_p (Mallows, 1973), AIC (Akaike, 1974) and BIC (Schwarz, 1978). Under certain regularity conditions, Wang et al. (2007) and Wang et al. (2009) established the selection consistency of BIC for the SCAD, and Zhang et al. (2010) showed the selection consistency of generalized information criterion (GIC) for the SCAD. Most of these criteria follow the route of minimizing the estimated prediction error or maximizing the posterior model probability. To the best of our knowledge, few criteria has been developed directly focusing on the selection of the informative variables.

This article proposes a tuning parameter selection criterion based on variable selection stability. The key idea is that if multiple samples are available from the same distribution, a good variable selection method should yield similar sets of informative variables that do not vary much from one sample to another. The similarity between two informative variable sets is measured by Cohen's kappa coefficient (Cohen, 1960), which adjusts the actual variable selection agreement relative to the possible agreement by chance. Similar stability measures have been studied in the context of cluster analysis (Ben-Hur et al., 2002; Wang, 2010) and variable selection (Meinshausen and Bühlmann, 2010). Whereas the stability selection method (Meinshausen and Bühlmann, 2010) also follows the idea of variable selection stability, it mainly focuses on selecting the informative variables as opposed to selecting the tuning parameters for any given variable selection methods. The effectiveness of the proposed selection criterion is demonstrated in a variety of simulated examples and a real application. More importantly, its asymptotic selection consistency is established, showing that the variable selection method with the selected tuning parameter would recover the truly informative variable set with probability tending to one.

The rest of the article is organized as follows. Section 2 briefly reviews the penalized regression models. Section 3 presents the idea of variable selection stability as well as the proposed kappa selection criterion. Section 4 establishes the asymptotic selection consistency of the kappa selection criterion. Simulation studies are given in Section 5, followed by a real application in Section 6. A brief discussion is provided in Section 7, and the Appendix is devoted to the technical proofs.

2. Penalized Least Squares Regression

Given that $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ are independent and identically distributed from some unknown joint distribution, we consider the linear regression model

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon = \sum_{j=1}^{p} \beta_j \mathbf{x}_{(j)} + \varepsilon,$$

where $\beta = (\beta_1, \cdots, \beta_p)^T$, $\mathbf{y} = (y_1, \cdots, y_n)^T$, $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)^T = (\mathbf{x}_{(1)}, \cdots, \mathbf{x}_{(p)})$ with $\mathbf{x}_i = (x_{i1}, \cdots, x_{ip})^T$ or $\mathbf{x}_{(j)} = (x_{1j}, \cdots, x_{nj})^T$, and $\varepsilon | \mathbf{X} \sim N(\mathbf{0}, \sigma^2 \mathbf{I}_n)$. When p is large, it is also assumed that only a small number of β_j's are nonzero, corresponding to the truly informative variables. In addition, both $\mathbf{y}$ and $\mathbf{x}_{(j)}$'s are centered, so the intercept can be omitted in the regression model.

The general framework of the penalized regression models can be formulated as

$$\underset{\beta}{\operatorname{argmin}} \ \frac{1}{n} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \sum_{j=1}^{p} p_\lambda(|\beta_j|), \tag{1}$$

where $\|\cdot\|$ is the Euclidean norm, and $p_\lambda(|\beta_j|)$ is a regularization term encouraging sparsity in β. Widely used regularization terms include the lasso penalty $p_\lambda(\theta) = \lambda\theta$ (Tibshirani, 1996), the SCAD penalty with $p'_\lambda(\theta) = \lambda(I(\theta \le \lambda) + \frac{(\gamma\lambda - \theta)_+}{(\gamma - 1)\lambda} I(\theta > \lambda))$ (Fan and Li, 2001), the adaptive lasso penalty $p_\lambda(\theta) = \lambda_j \theta = \lambda\theta / |\hat{\beta}_j|$ (Zou, 2006) with $\hat{\beta}_j$ being some initial estimate of β_j, and the truncated l_1-norm penalty $p_\lambda(\theta) = \lambda \min(1, \theta)$ (Shen et al., 2012).

With appropriately chosen λ_n, all the aforementioned regularization terms have been shown to be selection consistent. Here a penalty term is said to be selection consistent if the probability that the fitted regression model includes only the truly informative variables is tending to one, and λ is replaced by λ_n to emphasize its dependence on n in quantifying the asymptotic behaviors. In particular, Zhao and Yu (2006) showed that the lasso regression is selection consistent under the irrepresentable condition when $\sqrt{n}\lambda_n \to \infty$ and $\lambda_n \to 0$; Fan and Li (2001) showed that the SCAD is selection consistent when $\sqrt{n}\lambda_n \to \infty$ and $\lambda_n \to 0$; Zou (2006) showed that the adaptive lasso is selection consistent when $n\lambda_n \to \infty$ and $\sqrt{n}\lambda_n \to 0$; and Shen et al. (2012) showed that the truncated l_1-norm penalty is also selection consistent when λ_n satisfies a relatively more complex constraint.

Although the asymptotic order of λ_n is known to assure the selection consistency of the penalized regression models, it remains unclear how to appropriately select λ_n in finite sample so that the resultant model in (1) with the selected λ_n can achieve superior numerical performance and attain asymptotic selection consistency. Therefore, it is in demand to devise a tuning parameter selection criterion that can be employed by the penalized regression models so that their variable selection performance can be optimized.

3. Tuning via Variable Selection Stability

This section introduces the proposed tuning parameter selection criterion based on the concept of variable selection stability. The key idea is that if we repeatedly draw samples from the population and apply the candidate variable selection methods, a desirable method should produce the informative variable set that does not vary much from one sample to another. Clearly, variable selection stability is assumption free and can be used to tune any penalized regression model.

3.1 Variable Selection Stability

For simplicity, we denote the training sample as z^n. A base variable selection method $\Psi(z^n;\lambda)$ with a given training sample z^n and a tuning parameter λ yields a set of selected informative variables $\mathcal{A} \subset \{1,\cdots,p\}$, called the active set. When Ψ is applied to various training samples, different active sets can be produced. Supposed that two active sets $\mathcal{A}_1$ and $\mathcal{A}_2$ are produced, the agreement between $\mathcal{A}_1$ and $\mathcal{A}_2$ can be measured by Cohen's kappa coefficient (Cohen, 1960),

$$\kappa(\mathcal{A}_1,\mathcal{A}_2) = \frac{Pr(a)-Pr(e)}{1-Pr(e)}. \tag{2}$$

Here the relative observed agreement between $\mathcal{A}_1$ and $\mathcal{A}_2$ is $Pr(a) = (n_{11}+n_{22})/p$, and the hypothetical probability of chance agreement $Pr(e) = (n_{11}+n_{12})(n_{11}+n_{21})/p^2 + (n_{12}+n_{22})(n_{21}+n_{22})/p^2$, with $n_{11} = |\mathcal{A}_1 \cap \mathcal{A}_2|$, $n_{12} = |\mathcal{A}_1 \cap \mathcal{A}_2^c|$, $n_{21} = |\mathcal{A}_1^c \cap \mathcal{A}_2|$, $n_{22} = |\mathcal{A}_1^c \cap \mathcal{A}_2^c|$, and $|\cdot|$ being the set cardinality. Note that $-1 \le \kappa(\mathcal{A}_1,\mathcal{A}_2) \le 1$, where $\kappa(\mathcal{A}_1,\mathcal{A}_2) = 1$ when $\mathcal{A}_1$ and $\mathcal{A}_2$ are in complete agreement with $n_{12} = n_{21} = 0$, and $\kappa(\mathcal{A}_1,\mathcal{A}_2) = -1$ when $\mathcal{A}_1$ and $\mathcal{A}_2$ are in complete disagreement with $n_{11} = n_{22} = 0$ and $n_{12} = n_{21} = p/2$. For degenerate cases with $\mathcal{A}_1 = \mathcal{A}_2 = \emptyset$ or $\mathcal{A}_1 = \mathcal{A}_2 = \{1,\ldots,p\}$, we set $\kappa(\emptyset,\emptyset) = \kappa(\{1,\ldots,p\},\{1,\ldots,p\}) = -1$ under the assumption that the true model is sparse and containing at least one informative variable. As a consequence, the kappa coefficient in (2) is not suitable for evaluating the null model with no informative variable and the complete model with all variables. Based on (2), the variable selection stability is defined as follows.

Definition 1 *The variable selection stability of $\Psi(\cdot;\lambda)$ is defined as*

$$s(\Psi,\lambda,n) = E\Big(\kappa(\Psi(Z_1^n;\lambda),\Psi(Z_2^n;\lambda))\Big),$$

where the expectation is taken with respect to Z_1^n and Z_2^n, two independent and identically training samples of size n, and $\Psi(Z_1^n;\lambda)$ and $\Psi(Z_2^n;\lambda)$ are two active sets obtained by applying $\Psi(\cdot;\lambda)$ to Z_1^n and Z_2^n, respectively.

By definition, $-1 \le s(\Psi,\lambda,n) \le 1$, and large value of $s(\Psi,\lambda,n)$ indicates a stable variable selection method $\Psi(\cdot;\lambda)$. Note that the definition of $s(\Psi,\lambda,n)$ relies on the unknown population distribution, therefore it needs to be estimated based on the only available training sample in practice.

3.2 Kappa Selection Criterion

This section proposes an estimation scheme of the variable selection stability based on cross validation, and develops a kappa selection criterion to tune the penalized regression models by maximizing the estimated variable selection stability. Specifically, the training sample z^n is randomly partitioned into two subsets z_1^m and z_2^m with $m = \lfloor n/2 \rfloor$ for simplicity. The base variable selection method $\Psi(\cdot;\lambda)$ is applied to two subsets separately, and then two active sets $\widehat{\mathcal{A}}_{1\lambda}$ and $\widehat{\mathcal{A}}_{2\lambda}$ are obtained, and $s(\Psi,\lambda,m)$ is estimated as $\kappa(\widehat{\mathcal{A}}_{1\lambda},\widehat{\mathcal{A}}_{2\lambda})$. Furthermore, in order to reduce the estimation variability due to the splitting randomness, multiple data splitting can be conducted and the average estimated variable selection stability over all splittings is computed. The selected λ is then the one obtaining upper α_n quartile of the average estimated variable selection stability. The proposed kappa selection criterion is present as follows.

Algorithm 1 (kappa selection criterion) :

Step 1. Randomly partition $(\mathbf{x}_1,\cdots,\mathbf{x}_n)^T$ into two subsets $z_1^{*b}=(\mathbf{x}_1^{*b},\cdots,\mathbf{x}_m^{*b})^T$ and $z_2^{*b}=(\mathbf{x}_{m+1}^{*b},\cdots,\mathbf{x}_{2m}^{*b})^T$.

Step 2. Obtain $\widehat{\mathcal{A}}_{1\lambda}^{*b}$ and $\widehat{\mathcal{A}}_{2\lambda}^{*b}$ from $\Psi(z_1^{*b},\lambda)$ and $\Psi(z_2^{*b},\lambda)$ respectively, and estimate the variable selection stability of $\Psi(\cdot;\lambda)$ in the b-th splitting by

$$\hat{s}^{*b}(\Psi,\lambda,m)=\kappa(\widehat{\mathcal{A}}_{1\lambda}^{*b},\widehat{\mathcal{A}}_{2\lambda}^{*b}).$$

Step 3. Repeat *Steps 1-2* for B times. The average estimated variable selection stability of $\Psi(\cdot;\lambda)$ is then

$$\hat{s}(\Psi,\lambda,m)=B^{-1}\sum_{b=1}^{B}\hat{s}^{*b}(\Psi,\lambda,m).$$

Step 4. Compute $\hat{s}(\Psi,\lambda,m)$ for a sequence of λ's, and select

$$\hat{\lambda}=\min\Big\{\lambda:\frac{\hat{s}(\Psi,\lambda,m)}{\max_{\lambda'}\hat{s}(\Psi,\lambda',m)}\geq 1-\alpha_n\Big\}.$$

Note that the treatment in Step 4 is necessary since some informative variables may have relatively weak effect compared with others. A large value of λ may produce an active set that consistently overlooks the weakly informative variables, which leads to an underfitted model with large variable selection stability. To assure the asymptotic selection consistency, the thresholding value α_n in Step 4 needs to be small and converges to 0 as n grows. Setting $\alpha_n=0.1$ in the numerical experiments yields satisfactory performance based on our limited experience. Furthermore, the sensitivity study in Section 5.1 suggests that α_n has very little effect on the selection performance when it varies in a certain range. In Steps 1-3, the estimation scheme based on cross-validation can be replaced by other data re-sampling strategies such as bootstrap or random weighting, which do not reduce the sample size in estimating $\widehat{\mathcal{A}}_{1\lambda}^{*b}$ and $\widehat{\mathcal{A}}_{2\lambda}^{*b}$, but the independence between $\widehat{\mathcal{A}}_{1\lambda}^{*b}$ and $\widehat{\mathcal{A}}_{2\lambda}^{*b}$ will no longer hold.

The proposed kappa selection criterion shares the similar idea of variable selection stability with the stability selection method (Meinshausen and Bühlmann, 2010), but they differ in a number of ways. First, the stability selection method is a competitive variable selection method, which combines the randomized lasso regression and the bootstrap, and achieves superior variable selection performance. However, the kappa selection criterion can be regarded as a model selection criterion that is designed to select appropriate tuning parameters for any variable selection method. Second, despite of its robustness, the stability selection method still requires a number of tuning parameters. The authors proposed to select the tuning parameters via controlling the expected number of falsely selected variables. However, this criterion is less applicable in practice since the expected number of falsely selected variables can only be upper bounded by an expression involving various unknown quantities. On the contrary, the kappa selection criterion can be directly applied to select the tuning parameters for the stability selection method.

4. Asymptotic Selection Consistency

This section presents the asymptotic selection consistency of the proposed kappa selection criterion. Without loss of generality, we assume that only the first p_0 variables with $0<p_0<p$ are informative, and denote the truly informative variable set as $\mathcal{A}_T=\{1,\cdots,p_0\}$ and the uninformative variable set

as $\mathcal{A}_T^c = \{p_0+1, \cdots, p\}$. Furthermore, we denote $r_n \prec s_n$ if r_n converges to 0 at a faster rate than s_n, $r_n \sim s_n$ if r_n converges to 0 at the same rate as s_n, and $r_n \preceq s_n$ if r_n converges to 0 at a rate not slower than s_n.

4.1 Consistency with Fixed p

To establish the asymptotic selection consistency with fixed p, the following technical assumptions are made.

Assumption 1: There exist positive r_n and s_n such that the base variable selection method is selection consistent if $r_n \prec \lambda_n \prec s_n$. Let λ_n^* be such a tuning parameter with $r_n \prec \lambda_n^* \prec s_n$, then $P(\widehat{\mathcal{A}}_{\lambda_n^*} = \mathcal{A}_T) \geq 1-\varepsilon_n$ for some $\varepsilon_n \to 0$. In addition, for any positive constant λ_0, there exists positive $c_0(\lambda_0)$ such that, when n is sufficiently large,

$$P\Big(\bigcap_{\lambda_0 r_n \leq \lambda_n \leq \lambda_n^*} \{\widehat{\mathcal{A}}_{\lambda_n} = \mathcal{A}_T\}\Big) \geq 1 - c_0(\lambda_0), \tag{3}$$

where $c_0(\lambda_0)$ converges to 0 as $\lambda_0 \to \infty$.

Assumption 1 specifies an asymptotic working interval for λ_n within which the base variable selection method is selection consistent. Here the consistent rate ε_n is defined for λ_n^* only, and needs not hold uniformly over all λ_n with $r_n \prec \lambda_n \prec s_n$. Furthermore, (3) establishes an uniform lower bound for the probability of selecting the true model when λ_n is within the interval $(\lambda_0 r_n, \lambda_n^*)$.

Assumption 2: Given r_n in Assumption 1, for any positive constant λ_0, there exist ζ_n, $c_1(\lambda_0)$ and $c_2(\lambda_0)$ such that, when n is sufficiently large,

$$\min_{j \in \mathcal{A}_T} P\Big(\bigcap_{r_n^{-1}\lambda_n \leq \lambda_0} \{j \in \widehat{\mathcal{A}}_{\lambda_n}\}\Big) \geq 1 - \zeta_n, \tag{4}$$

$$\min_{j \in \mathcal{A}_T^c} P\Big(\bigcap_{r_n^{-1}\lambda_n \leq \lambda_0} \{j \in \widehat{\mathcal{A}}_{\lambda_n}\}\Big) \geq c_1(\lambda_0), \tag{5}$$

$$\max_{j \in \mathcal{A}_T^c} P\Big(\bigcap_{r_n^{-1}\lambda_n \geq \lambda_0} \{j \notin \widehat{\mathcal{A}}_{\lambda_n}\}\Big) \geq c_2(\lambda_0), \tag{6}$$

where $\zeta_n \to 0$ as $n \to \infty$, $c_1(\lambda_0)$ and $c_2(\lambda_0)$ are positive and only depend on λ_0, and $c_1(\lambda_0) \to 1$ as $\lambda_0 \to 0$.

Assumption 2 implies that if λ_n converges to 0 faster than r_n, the base variable selection method will select all the variables asymptotically, and when λ_n converges to 0 at the same rate of r_n, the base variable selection method will select any noise variable with an asymptotically positive probability. The inequalities (4)-(6) also establish uniform lower bounds for various probabilities of selecting informative variables or noise variables.

Assumptions 1 and 2 are mild in that they are satisfied by many popular variable selection methods. For instance, Lemma 2 in the online supplementary material shows that Assumptions 1 and 2 are satisfied by the lasso regression, the adaptive lasso, and the SCAD. The assumptions can also be verified for other methods such as the elastic-net (Zou and Hastie, 2005), the adaptive elastic net (Zou and Zhang, 2009), the group lasso (Yuan and Lin, 2006), and the adaptive group lasso (Wang and Leng, 2008).

Given that the base variable selection method is selection consistent with appropriately selected λ_n's, Theorem 1 shows that the proposed kappa selection criterion is able to identify such λ_n's.

Theorem 1 *Under Assumptions 1 and 2, any variable selection method in (1) with $\hat{\lambda}_n$ selected as in Algorithm 1 with $\alpha_n \succ \varepsilon_n$ is selection consistent. That is,*

$$\lim_{n\to\infty}\lim_{B\to\infty} P(\widehat{\mathcal{A}}_{\hat{\lambda}_n} = \mathcal{A}_T) = 1.$$

Theorem 1 claims the asymptotic selection consistency of the proposed kappa selection criterion when p is fixed. That is, with probability tending to one, the selected active set by the resultant variable selection method with tuning parameter $\hat{\lambda}_n$ contains only the truly informative variables. As long as α_n converges to 0 not too fast, the kappa selection criterion is guaranteed to be consistent. Therefore, the value of α_n is expected to have little effect on the performance of the kappa selection criterion, which agrees with the sensitivity study in Section 5.1.

4.2 Consistency with Diverging p_n

In high-dimensional data analysis, it is of interest to study the asymptotic behavior of the proposed kappa selection criterion with diverging p_n, where size of truly informative set p_{0n} may also diverge with n. To accommodate the diverging p_n scenario, the technical assumptions are modified as follows.

Assumption 1a: There exist positive r_n and s_n such that the base variable selection method is selection consistent if $r_n \prec \lambda_n \prec s_n$. Let λ_n^* be such a tuning parameter with $r_n \prec \lambda_n^* \prec s_n$, then $P(\widehat{\mathcal{A}}_{\lambda_n^*} = \mathcal{A}_T) \geq 1-\varepsilon_n$ for some $\varepsilon_n \to 0$. In addition, for any positive constant λ_0, there exists positive $c_{0n}(\lambda_0)$ such that, when n is sufficiently large,

$$P\Big(\bigcap_{\lambda_0 r_n \leq \lambda_n \leq \lambda_n^*} \{\widehat{\mathcal{A}}_{\lambda_n} = \mathcal{A}_T\}\Big) \geq 1 - c_{0n}(\lambda_0), \tag{7}$$

where $\lim_{\lambda_0\to\infty}\lim_{n\to\infty} c_{0n}(\lambda_0) \to 0$.

Assumption 2a: Given r_n in Assumption 1a, for any positive constant λ_0, there exist ζ_n, $c_{1n}(\lambda_0)$ and $c_{2n}(\lambda_0)$ such that, when n is sufficiently large,

$$\min_{j\in\mathcal{A}_T} P\Big(\bigcap_{r_n^{-1}\lambda_n \leq \lambda_0} \{j \in \widehat{\mathcal{A}}_{\lambda_n}\}\Big) \geq 1-\zeta_n, \tag{8}$$

$$\min_{j\in\mathcal{A}_T^c} P\Big(\bigcap_{r_n^{-1}\lambda_n \leq \lambda_0} \{j \in \widehat{\mathcal{A}}_{\lambda_n}\}\Big) \geq c_{1n}(\lambda_0), \tag{9}$$

$$\max_{j\in\mathcal{A}_T^c} P\Big(\bigcap_{r_n^{-1}\lambda_n \geq \lambda_0} \{j \notin \widehat{\mathcal{A}}_{\lambda_n}\}\Big) \geq c_{2n}(\lambda_0), \tag{10}$$

where ζ_n satisfies $p_n\zeta_n \to 0$ as $n\to\infty$, $c_{1n}(\lambda_0)$ and $c_{2n}(\lambda_0)$ are positive and may depend on n and λ_0, and $\lim_{\lambda_0\to 0}\lim_{n\to\infty} c_{1n}(\lambda_0) = 1$.

Theorem 2 *Under Assumptions 1a and 2a, any variable selection method in (1) with $\hat{\lambda}_n$ selected as in Algorithm 1 with $\min(p_n(1-\tilde{c}_{1n}), p_n^{-1}c_{1n}c_{2n}) \succ \alpha_n \succ \varepsilon_n$ is selection consistent, where $\tilde{c}_{1n} = \sup_{\lambda_0} c_{1n}(\lambda_0)$, $c_{1n} = \inf_{\lambda_0} c_{1n}(\lambda_0)$, and $c_{2n} = \inf_{\lambda_0} c_{2n}(\lambda_0)$.*

Theorem 2 shows the asymptotic selection consistency of the proposed kappa selection criterion with satisfied α_n for diverging p_n, where the diverging speed of p_n is bounded as in Theorem 2 and

depends on the base variable selection method. Lemma 3 in the online supplementary material shows that (7)-(10) in Assumptions 1a and 2a are satisfied by the lasso regression. However, it is generally difficult to verify Assumptions 1a and 2a for other popular variable selection algorithms (Fan and Peng, 2004; Huang and Xie, 2007; Huang et al., 2008), as the convergence rates in both assumptions are not explicitly specified.

5. Simulations

This section examines the effectiveness of the proposed kappa selection criterion in simulated examples. Its performance is compared against a number of popular competitors, including Mallows' C_p (C_p), BIC, 10-fold cross validation (CV), and generalized cross validation (GCV). Their formulations are given as follows,

$$\begin{aligned} C_p(\lambda) &= \frac{SSE_\lambda}{\hat{\sigma}^2} - n + 2\widehat{df}, \qquad (11) \\ BIC(\lambda) &= \log\Big(\frac{SSE_\lambda}{n}\Big) + \frac{\log(n)\widehat{df}}{n}, \\ CV(\lambda) &= \sum_{s=1}^{10} \sum_{(y_k,x_k)\in T^{-s}} \Big(y_k - \mathbf{x}_k^T \hat{\beta}^{(s)}(\lambda)\Big)^2, \qquad (12) \\ GCV(\lambda) &= \frac{SSE_\lambda}{n(1-\widehat{df}/n)^2}, \end{aligned}$$

where $SSE_\lambda = \|\mathbf{y} - \mathbf{X}\hat{\beta}(\lambda)\|^2$, $\widehat{df}$ is estimated as the number of nonzero variables in $\hat{\beta}(\lambda)$ (Zou et al., 2007), and $\hat{\sigma}^2$ in (11) is estimated based on the saturated model. In (12), T^s and T^{-s} are the training and validation sets in CV, and $\hat{\beta}^{(s)}(\lambda)$ is the estimated β using the training set T^s and tuning parameter λ. The optimal $\hat{\lambda}$ is then selected as the one that minimizes the corresponding $C_p(\lambda)$, $BIC(\lambda)$, $CV(\lambda)$, or $GCV(\lambda)$, respectively.

To assess the performance of each selection criterion, we report the percentage of selecting the true model over all replicates, as well as the number of correctly selected zeros and incorrectly selected zeros in $\hat{\beta}(\hat{\lambda})$. The final estimator $\hat{\beta}(\hat{\lambda})$ is obtained by refitting the standard least squares regression based only on the selected informative variables. We then compare the prediction performance through the relative prediction error $RPE = E(\mathbf{x}^T\hat{\beta}(\hat{\lambda}) - \mathbf{x}^T\beta)^2/\sigma^2$ (Zou, 2006).

5.1 Scenario I: Fixed p

The simulated data sets $(\mathbf{x}_i, y_i)_{i=1}^n$ are generated from the model

$$y = \mathbf{x}^T\beta + \sigma\varepsilon = \sum_{j=1}^{8} \mathbf{x}_{(j)}\beta_j + \sigma\varepsilon,$$

where $\beta = (3, 1.5, 0, 0, 2, 0, 0, 0)^T$, $\sigma = 1$, $\mathbf{x}_{(j)}$ and ε are generated from standard normal distribution, and the correlation between $\mathbf{x}_{(i)}$ and $\mathbf{x}_{(j)}$ is set as $0.5^{|i-j|}$. This example has been commonly used in literature, including Tibshirani (1996), Fan and Li (2001), and Wang et al. (2007).

For comparison, we set $n = 40$, 60 or 80 and implement the lasso regression, the adaptive lasso and the SCAD as the base variable selection methods. The lasso regression and the adaptive lasso

are implemented by package 'lars' (Efron et al., 2004) and the SCAD is implemented by package 'ncvreg' (Breheny and Huang, 2011) in R. The tuning parameter λ's are selected via each selection criterion, optimized through a grid search over 100 points $\{10^{-2+4l/99};\ l = 0,\ldots,99\}$. The number of splittings B for the kappa selection criterion is 20. Each simulation is replicated 100 times, and the percentages of selecting the true active set, the average numbers of correctly selected zeros (C) and incorrectly selected zeros (I), and the relative prediction errors (RPE) are summarized in Tables 1-2 and Figure 1.

n	Penalty	Ks	Cp	BIC	CV	GCV
40	Lasso	**0.63**	0.16	0.26	0.09	0.16
	Ada lasso	**0.98**	0.53	0.72	0.63	0.52
	SCAD	**0.98**	0.55	0.78	0.76	0.52
60	Lasso	**0.81**	0.16	0.32	0.14	0.17
	Ada lasso	**0.99**	0.52	0.84	0.65	0.52
	SCAD	**1**	0.58	0.86	0.76	0.56
80	Lasso	**0.89**	0.16	0.38	0.08	0.16
	Ada lasso	**0.99**	0.56	0.86	0.77	0.56
	SCAD	**0.99**	0.62	0.89	0.75	0.61

Table 1: The percentages of selecting the true active set for various selection criteria in simulations of Section 5.1. Here 'Ks', 'Cp', 'BIC', 'CV' and 'GCV' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

		Ks	Ks	Cp	Cp	BIC	BIC	CV	CV	GCV	GCV
n	Penalty	C	I	C	I	C	I	C	I	C	I
40	Lasso	**4.58**	0.01	3.26	0	3.60	0	2.66	0	3.25	0
	Ada lasso	**4.98**	0	4.16	0	4.54	0	4.25	0	4.15	0
	SCAD	**4.99**	0.01	4.11	0	4.59	0	4.39	0	4.06	0
60	Lasso	**4.80**	0	3.12	0	3.91	0	2.85	0	3.13	0
	Ada lasso	**4.99**	0	4.17	0	4.80	0	4.35	0	4.17	0
	SCAD	**5**	0	4.15	0	4.79	0	4.37	0	4.12	0
80	Lasso	**4.88**	0	3.01	0	4.02	0	2.66	0	3	0
	Ada lasso	**4.99**	0	4.19	0	4.80	0	4.49	0	4.19	0
	SCAD	**4.99**	0	4.23	0	4.83	0	4.45	0	4.22	0

Table 2: The average numbers of correctly selected zeros (C) and incorrectly selected zeros (I) for various selection criteria in simulations of Section 5.1. Here 'Ks', 'Cp', 'BIC', 'CV' and 'GCV' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

Evidently, the proposed kappa selection criterion delivers superior performance against its competitors in terms of both variable selection accuracy and relative prediction error. As shown in Table 1, the kappa selection criterion has the largest probability of choosing the true active set and consistently outperforms other selection criteria, especially when the lasso regression is used as the base variable selection method. As the sample size n increases, the percentage of selecting the true active set is also improving, which supports the selection consistency in Section 4.

Table 2 shows that the kappa selection criterion yields the largest number of correctly selected zeros in all scenarios, and it yields almost perfect performance for the adaptive lasso and the SCAD. In addition, all selection criteria barely select any incorrect zeros, whereas the kappa selection criterion is relatively more aggressive in that it has small chance to shrink some informative variables to zeros when sample size is small. All other criteria tend to be conservative and include some uninformative variables, so the numbers of correctly selected zeros are significantly less than 5.

Besides the superior variable selection performance, the kappa selection criterion also delivers accurate prediction performance and yields small relative prediction error as displayed in Figure 1. Note that other criteria, especially C_p and GCV, produce large relative prediction errors, which could be due to their conservative selection of the informative variables.

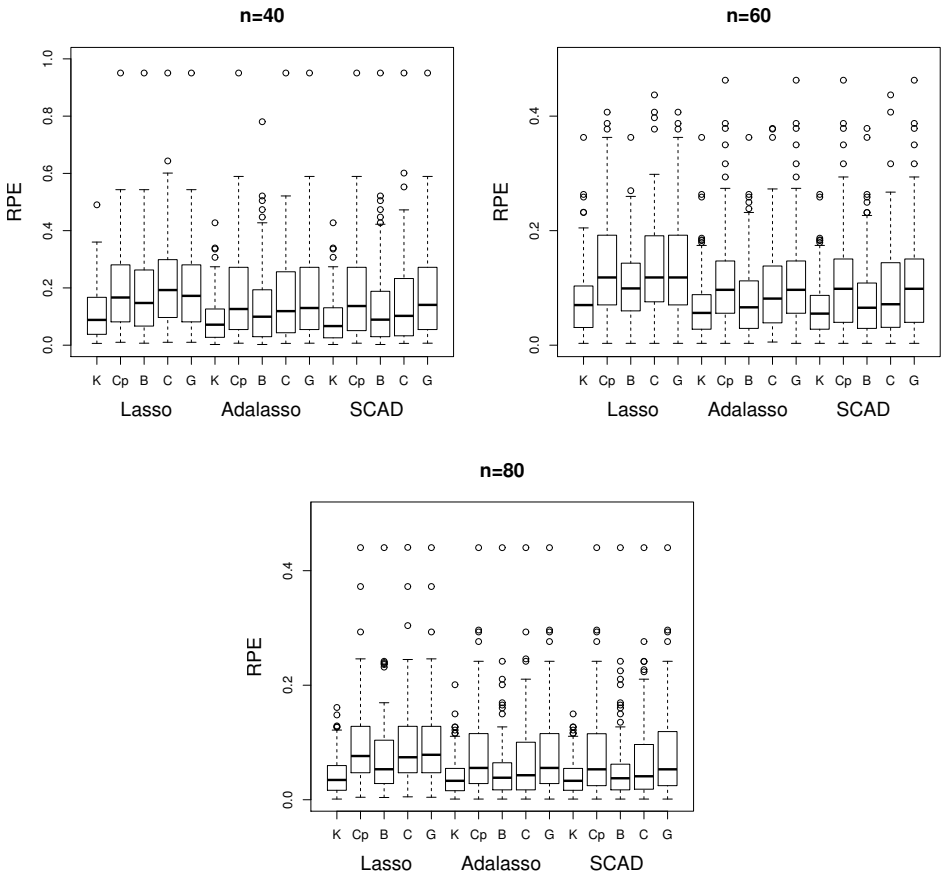


Figure 1: Relative prediction errors (RPE) for various selection criteria in simulations of Section 5.1. Here 'K', 'Cp', 'B', 'C' and 'G' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

To illustrate the effectiveness of the kappa selection criterion, we randomly select one replication with $n = 40$ and display the estimated variable selection stability as well as the results of detection and sparsity for various λ's for the lasso regression. The detection is defined as the per-

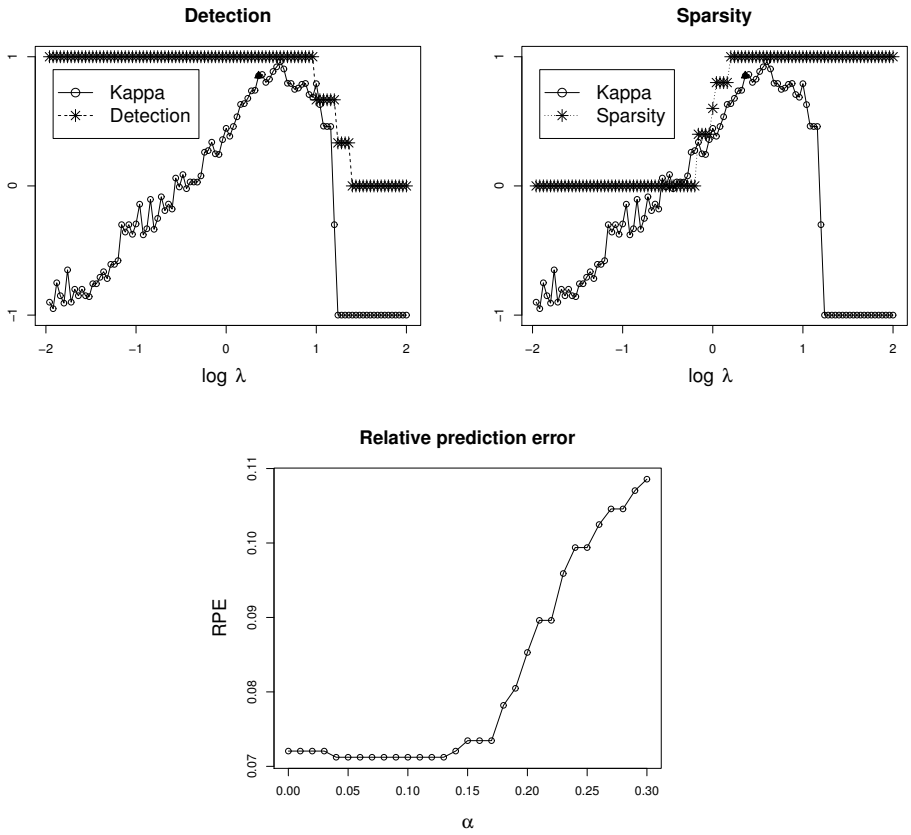


Figure 2: The detection and sparsity of the lasso regression with the kappa selection criterion are shown on the top and the sensitivity of α to the relative prediction error is shown on the bottom. The optimal $\log(\lambda)$ selected by the kappa selection criterion is denoted as the filled triangle in the detection and sparsity plots.

centage of selecting the truly informative variables, and the sparsity is defined as the percentage of excluding the truly uninformative variables. Figure 2 illustrates the clear relationship between the variable selection stability and the values of detection and sparsity. More importantly, the selection performance of the kappa selection criterion is very stable against α_n when it is small. Specifically, we apply the kappa selection criterion on the lasso regression for $\alpha_n = \{\frac{l}{100};\ l = 0,\ldots,30\}$ and compute the corresponding average RPE over 100 replications. As shown in the last panel of Figure 2, the average RPE's are almost the same for $\alpha_n \in (0, 0.13)$, which agrees with the theoretical result in Section 4.

5.2 Scenario II: Diverging p_n

To investigate the effects of the noise level and the dimensionality, we compare all the selection criteria in the diverging p_n scenario with a similar simulation model as in Scenario I, except that $\beta = (5,4,3,2,1,0,\cdots,0)^T$, $p_n = [\sqrt{n}]$, and $\sigma = 1$ or 6. More specifically, 8 cases are examined:

$n = 100,\ p_n = 10$; $n = 200,\ p_n = 14$; $n = 400,\ p_n = 20$; and $n = 800,\ p_n = 28$, with $\sigma = 1$ or 6 respectively. Note that when $\sigma = 6$, the truly informative variables are much more difficult to detect due to the increased noise level. The percentages of selecting the true active set, the average numbers of correctly selected zeros (C) and incorrectly selected zeros (I), and the relative prediction errors (RPE) are summarized in Tables 3-4 and Figures 3-4.

n	p_n	Penalty	Ks	Cp	BIC	CV	GCV
					$\sigma = 1$		
		Lasso	**0.98**	0.17	0.43	0.10	0.17
100	10	Ada lasso	**0.99**	0.48	0.86	0.74	0.47
		SCAD	**0.97**	0.47	0.92	0.82	0.47
		Lasso	**1**	0.11	0.49	0.07	0.11
200	14	Ada lasso	**1**	0.38	0.90	0.66	0.38
		SCAD	**1**	0.46	0.93	0.73	0.47
		Lasso	**1**	0.09	0.53	0.04	0.09
400	20	Ada lasso	**1**	0.34	0.93	0.73	0.33
		SCAD	**1**	0.43	0.98	0.75	0.43
		Lasso	**1**	0.11	0.51	0.04	0.11
800	28	Ada lasso	**1**	0.30	0.96	0.74	0.29
		SCAD	**1**	0.46	0.99	0.71	0.46
					$\sigma = 6$		
		Lasso	**0.35**	0.14	0.31	0.11	0.15
100	10	Ada lasso	**0.21**	0.15	0.18	0.11	0.15
		SCAD	**0.17**	0.07	0.12	0.12	0.07
		Lasso	**0.52**	0.10	0.39	0.08	0.09
200	14	Ada lasso	**0.40**	0.18	0.30	0.16	0.18
		SCAD	**0.24**	0.09	0.15	0.13	0.09
		Lasso	**0.77**	0.10	0.47	0.04	0.10
400	20	Ada lasso	0.53	0.22	**0.57**	0.24	0.19
		SCAD	**0.40**	0.13	0.30	0.13	0.13
		Lasso	**0.82**	0.07	0.51	0.04	0.06
800	28	Ada lasso	**0.68**	0.20	0.66	0.37	0.20
		SCAD	**0.46**	0.21	0.39	0.17	0.21

Table 3: The percentages of selecting the true active set for various selection criteria in simulations of Section 5.2. Here 'Ks', 'Cp', 'BIC', 'CV' and 'GCV' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

In the low noise case with $\sigma = 1$, the proposed kappa selection criterion outperforms other competitors in both variable selection and prediction performance. As illustrated in Tables 3-4, the kappa selection criterion delivers the largest percentage of selecting the true active set among all the selection criteria, and achieves perfect variable selection performance for all the variable selection methods when $n \geq 200$. Furthermore, as shown in Figure 3, the kappa selection criterion yields the smallest relative prediction error across all cases.

As the noise level increases to $\sigma = 6$, the kappa selection criterion still delivers the largest percentage of selecting the true active set among all scenarios except for the adaptive lasso with $n = 400$, where the percentage is slightly smaller than that of BIC. As shown in Table 4, the kappa

			Ks	Ks	Cp	Cp	BIC	BIC	CV	CV	GCV	GCV
n	p_n	Penalty	C	I	C	I	C	I	C	I	C	I
			$\sigma = 1$									
		Lasso	**5**	0.02	3.25	0	4.20	0	2.95	0	3.25	0
100	10	Ada lasso	**5**	0.01	4.23	0	4.84	0	4.48	0	4.21	0
		SCAD	**5**	0.03	4.12	0	4.91	0	4.67	0	4.15	0
		Lasso	**9**	0	6.18	0	8.26	0	5.62	0	6.18	0
200	14	Ada lasso	**9**	0	7.50	0	8.87	0	8.24	0	7.50	0
		SCAD	**9**	0	7.43	0	8.91	0	8.26	0	7.47	0
		Lasso	**15**	0	11.29	0	14.23	0	10.56	0	11.29	0
400	20	Ada lasso	**15**	0	12.93	0	14.92	0	14.28	0	12.91	0
		SCAD	**15**	0	12.67	0	14.98	0	14.21	0	12.64	0
		Lasso	**23**	0	18.49	0	22.27	0	18.20	0	18.63	0
800	28	Ada lasso	**23**	0	20.31	0	22.94	0	22.07	0	20.23	0
		SCAD	**23**	0	20.21	0	22.99	0	21.95	0	20.21	0
			$\sigma = 6$									
		Lasso	**4.76**	0.57	3.27	0.24	4.31	0.35	3.09	0.20	3.28	0.24
100	10	Ada lasso	4.57	0.77	3.81	0.54	**4.62**	0.85	3.31	0.49	3.84	0.54
		SCAD	**4.88**	1.22	3.63	0.56	4.37	0.94	3.52	0.58	3.65	0.56
		Lasso	**8.93**	0.43	6.20	0.08	8.32	0.21	5.79	0.07	6.22	0.08
200	14	Ada lasso	**8.72**	0.55	7.28	0.32	8.69	0.56	7.34	0.37	7.26	0.32
		SCAD	**9**	0.95	7.07	0.37	8.37	0.63	7.25	0.44	7.07	0.37
		Lasso	**14.98**	0.21	11.46	0.03	14.21	0.07	10.60	0.03	11.45	0.03
400	20	Ada lasso	**14.88**	0.40	12.24	0.09	14.80	0.30	12.93	0.15	12.16	0.09
		SCAD	**15**	0.67	11.97	0.13	14.65	0.51	12.66	0.23	11.88	0.12
		Lasso	**22.99**	0.17	18.65	0.01	22.27	0.01	18.14	0.01	18.68	0.01
800	28	Ada lasso	**22.96**	0.29	19.84	0.02	22.71	0.16	21.19	0.04	19.71	0.02
		SCAD	**23**	0.55	19.55	0.04	22.73	0.37	20.42	0.11	19.47	0.04

Table 4: The average numbers of correctly selected zeros (C) and incorrectly selected zeros (I) for various selection criteria in simulations of Section 5.2. Here 'Ks', 'Cp', 'BIC', 'CV' and 'GCV' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

selection criterion yields the largest number of correctly selection zeros. However, it has relatively higher chance of shrinking the fifth informative variable to zero, while the chance is diminishing as n increases. This phenomenon is also present for BIC. Considering the smaller relative prediction errors achieved by the kappa selection criterion and BIC, these two criteria tend to produce sparser models with satisfactory prediction performance. In practice, if false negatives are of concern, one can increase the thresholding value α_n in the kappa selection criterion, to allow higher tolerance of instability and hence decrease the chance of claiming false negatives. In addition, as shown in Figure 4, the kappa selection criterion yields the smallest relative prediction error for the lasso regression and the adaptive lasso among all scenarios, whereas the advantage is considerably less significant for the SCAD. This is somewhat expected as the SCAD is sensitive to the noise level (Zou, 2006), which may lead to inaccurate estimation of the variable selection stability.

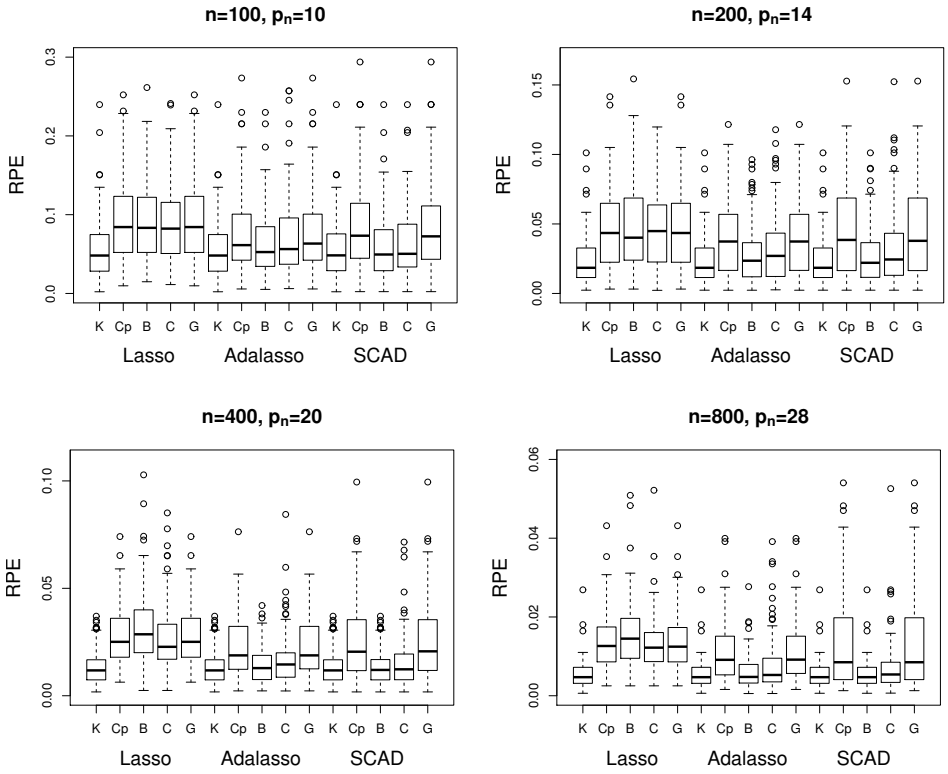


Figure 3: Relative prediction errors (RPE) for various selection criteria in Scenario 2 with $\sigma = 1$. Here 'K', 'Cp', 'B', 'C' and 'G' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

6. Real Application

In this section, we apply the kappa selection criterion to the prostate cancer data (Stamey et al., 1989), which were used to study the relationship between the level of log(prostate specific antigen) ($lpsa$) and a number of clinical measures. The data set consisted of 97 patients who had received a radical prostatectomy, and eight clinical measures were log(cancer volume) ($lcavol$), log(prostate weight) ($lweight$), age, log(benign prostaic hyperplasia amount) ($lbph$), seminal vesicle invasion (svi), log(capsular penetration) (lcp), Gleason score ($gleason$) and percentage Gleason scores 4 or 5 ($pgg45$).

The data set is randomly split into two halves: a training set with 67 patients and a test set with 30 patients. Similarly as in the simulated examples, the tuning parameter λ's are selected through a grid search over 100 grid points $\{10^{-2+4l/99};\ l = 0,\ldots,99\}$ on the training set. Since it is unknown whether the clinical measures are truly informative or not, the performance of all the selection criteria are compared by computing their corresponding prediction errors on the test set in Table 5.

As shown in Table 5, the proposed kappa selection criterion yields the sparsest model and achieves the smallest prediction error for the lasso regression and the SCAD, while the predic-

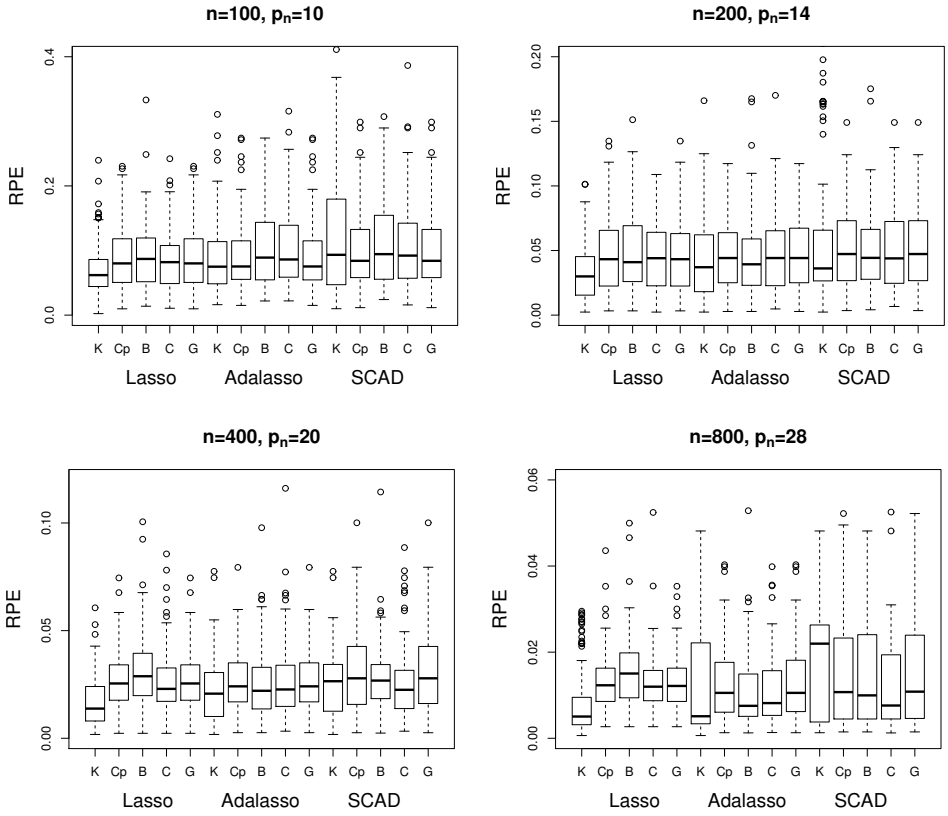


Figure 4: Relative prediction errors (RPE) for various selection criteria in Scenario 2 with $\sigma = 6$. Here 'K', 'Cp', 'B', 'C' and 'G' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

	Penalty	Ks	Cp	BIC	CV	GCV
Active	Lasso	1,2,4,5	1,2,3,4,5,6,7,8	1,2,4,5	1,2,3,4,5,7,8	1,2,3,4,5,6,7,8
Set	Ada lasso	1,2,5	1,2,3,4,5	1,2,3,4,5	1,2,3,4,5,6,7,8	1,2,3,4,5
	SCAD	1,2,4,5	1,2,3,4,5	1,2,3,4,5	1,2,3,4,5,6,7,8	1,2,3,4,5
	Lasso	**0.734**	0.797	**0.734**	0.807	0.797
PE	Ada lasso	0.806	0.825	0.825	**0.797**	0.825
	SCAD	**0.734**	0.825	0.825	0.797	0.825

Table 5: The selected active sets and the prediction errors (PE) for various selection criteria in the prostate cancer example. Here 'Ks', 'Cp', 'BIC', 'CV' and 'GCV' represent the kappa selection criterion, Mallows' C_p, BIC, CV and GCV, respectively.

tion error for the adaptive lasso is comparable to the minima. Specifically, the lasso regression and the SCAD with the kappa selection criterion include *lcavol*, *lweight*, *lbph* and *svi* as the informative variables, and the adaptive lasso with the kappa selection criterion selects only *lcavol*, *lweight*

and *svi* as the informative variables. As opposed to the sparse regression models produced by other selection criteria, the variable *age* is excluded by the kappa selection criterion for all base variable selection methods, which agrees with the findings in Zou and Hastie (2005).

7. Discussion

This article proposes a tuning parameter selection criterion based on the concept of variable selection stability. Its key idea is to select the tuning parameter so that the resultant variable selection method is stable in selecting the informative variables. The proposed criterion delivers superior numerical performance in a variety of experiments. Its asymptotic selection consistency is also established for both fixed and diverging dimensions. Furthermore, it is worth pointing out that the idea of stability is general and can be naturally extended to a broader framework of model selection, such as the penalized nonparametric regression (Xue et al., 2010) and the penalized clustering (Sun et al., 2012).

8. Supplementary Material

Lemmas 2 and 3 and their proofs are provided as online supplementary material for this article.

Acknowledgments

The authors thank the Action Editor and three referees for their constructive comments and suggestions, which have led to a significantly improved paper.

Appendix A.

The lemma stated below shows that if a variable selection method is selection consistent and $\varepsilon_n \prec \alpha_n$, then its variable selection stability converges to 1 in probability.

Lemma 3 *Let λ_n^* be as defined in Assumption 1. For any α_n,*

$$P\Big(\hat{s}(\Psi,\lambda_n^*,m) \geq 1-\alpha_n\Big) \geq 1-2\varepsilon_n/\alpha_n.$$

Proof of Lemma 3: We denote $\widehat{\mathcal{A}}^{*b}_{1\lambda_n^*}$ and $\widehat{\mathcal{A}}^{*b}_{2\lambda_n^*}$ as the corresponding active sets obtained from two sub-samples at the b-th random splitting. Then the estimated variable selection stability based on the b-th splitting can be bounded as

$$P\Big(\hat{s}^{*b}(\Psi,\lambda_n^*,m)=1\Big) = P\Big(\widehat{\mathcal{A}}^{*b}_{1\lambda_n^*} = \widehat{\mathcal{A}}^{*b}_{2\lambda_n^*}\Big) \geq P\Big(\widehat{\mathcal{A}}^{*b}_{1\lambda_n^*} = \mathcal{A}_T\Big)^2 \geq (1-\varepsilon_n)^2 \geq 1-2\varepsilon_n.$$

By the fact that $0 \leq \hat{s}^{*b}(\Psi,\lambda_n^*,n) \leq 1$, we have

$$E\Big(\hat{s}(\Psi,\lambda_n^*,m)\Big) = E\Big(B^{-1}\sum_{b=1}^{B}\hat{s}^{*b}(\Psi,\lambda_n^*,m)\Big) \geq 1-2\varepsilon_n,$$

and $0 \le \hat{s}(\Psi, \lambda_n^*, n) \le 1$. Finally, Markov inequality yields that

$$P\Big(1 - \hat{s}(\Psi, \lambda_n^*, m) \ge \alpha_n\Big) \le \frac{E\Big(1 - \hat{s}(\Psi, \lambda_n^*, m)\Big)}{\alpha_n} \le \frac{2\varepsilon_n}{\alpha_n},$$

which implies the desired result immediately. ■

Proof of Theorem 1: We first show that for any $\varepsilon > 0$,

$$\lim_{n\to\infty} P\Big(\hat{\lambda}_n > \lambda_n^* \text{ or } r_n^{-1}\hat{\lambda}_n \le 1/\varepsilon\Big) = 0.$$

Denote $\Omega_1 = \{\lambda_n : \lambda_n > \lambda_n^*\}$, $\Omega_2 = \{\lambda_n : r_n^{-1}\lambda_n \le \tau\}$ and $\Omega_3 = \{\lambda_n : \tau \le r_n^{-1}\lambda_n \le 1/\varepsilon\}$, where $\tau < 1/\varepsilon$, $c_1(\tau) \ge 1 - 1/p$. It then suffices to show that for any $\varepsilon > 0$,

$$P\Big(\hat{\lambda}_n \in \Omega_1 \cup \Omega_2 \cup \Omega_3\Big) \to 0.$$

First, the definition of $\hat{\lambda}_n$ and Lemma 1 imply that

$$P(\hat{\lambda}_n \le \lambda_n^*) \ge P\Big(\frac{\hat{s}(\Psi, \lambda_n^*, m)}{\max_\lambda \hat{s}(\Psi, \lambda, m)} \ge 1 - \alpha_n\Big) \ge P\Big(\hat{s}(\Psi, \lambda_n^*, m) \ge 1 - \alpha_n\Big) \ge 1 - \frac{2\varepsilon_n}{\alpha_n}.$$

This, together with $\varepsilon_n \prec \alpha_n$, yields that

$$P\Big(\hat{\lambda}_n \in \Omega_1\Big) = P(\hat{\lambda}_n > \lambda_n^*) \le \frac{2\varepsilon_n}{\alpha_n} \to 0.$$

Next, the definition of $\hat{\lambda}_n$ implies that

$$\hat{s}(\Psi, \hat{\lambda}_n, m) \ge (1 - \alpha_n) \max_\lambda \hat{s}(\Psi, \lambda, m) \ge (1 - \alpha_n)\hat{s}(\Psi, \lambda_n^*, m).$$

This, together with Lemma 1, leads to

$$\begin{aligned} P\Big(\hat{s}(\Psi, \hat{\lambda}_n, m) \ge 1 - 2\alpha_n\Big) &\ge P\Big(\hat{s}(\Psi, \hat{\lambda}_n, m) \ge (1 - \alpha_n)^2\Big) \\ &\ge P\Big(\hat{s}(\Psi, \lambda_n^*, m) \ge 1 - \alpha_n\Big) \ge 1 - \frac{2\varepsilon_n}{\alpha_n}, \end{aligned}$$

and hence when $\varepsilon_n \prec \alpha_n$,

$$P\Big(\hat{s}(\Psi, \hat{\lambda}_n, m) \ge 1 - 2\alpha_n\Big) \to 1.$$

Therefore, to show $P(\hat{\lambda}_n \in \Omega_2) \to 0$, it suffices to show

$$P\Big(\sup_{\lambda_n \in \Omega_2} \hat{s}(\Psi, \lambda_n, m) < 1 - 2\alpha_n\Big) \to 1. \tag{13}$$

But Assumption 2 implies that for any $j \in \mathcal{A}_T^c$ and $j_1 \in \mathcal{A}_T$, we have

$$P\Big(j \in \bigcap_{\lambda_n \in \Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) \ge c_1(\tau) \ge 1 - \frac{1}{p} \text{ and } P\Big(j_1 \in \bigcap_{\lambda_n \in \Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) \ge 1 - \zeta_n.$$

It implies that

$$\begin{aligned}
&\lim_{n\to\infty} P\Big(\{1,\ldots,p\} \in \bigcap_{\lambda_n\in\Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) \\
\geq\ &\lim_{n\to\infty} 1 - \sum_{j\in\mathcal{A}_T^c} P\Big(j \notin \bigcap_{\lambda_n\in\Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) - \sum_{j_1\in\mathcal{A}_T} P\Big(j_1 \notin \bigcap_{\lambda_n\in\Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) \\
\geq\ &\lim_{n\to\infty} 1 - \frac{p-p_0}{p} - p_0\zeta_n = \frac{p_0}{p} > 0.
\end{aligned}$$

Since $\{1,\ldots,p\} \in \bigcap_{\lambda_n\in\Omega_2} \widehat{\mathcal{A}}^{*b}_{\lambda_n}$ implies $\sup_{\lambda_n\in\Omega_2} \hat{s}^{*b}(\Psi,\lambda_n,m) = -1$, then

$$\lim_{n\to\infty} E\Big(\sup_{\lambda_n\in\Omega_2} \hat{s}^{*b}(\Psi,\lambda_n,m)\Big) \leq 1 - \lim_{n\to\infty} P\Big(\sup_{\lambda_n\in\Omega_2} \hat{s}^{*b}(\Psi,\lambda_n,m) = -1\Big) \leq 1 - \frac{p_0}{p}.$$

In addition, the strong law of large number for U-statistics (Hoeffding, 1961) implies that

$$B^{-1}\sum_{b=1}^{B} \sup_{\lambda_n\in\Omega_2} \hat{s}^{*b}(\Psi,\lambda_n,m) \xrightarrow{a.s.} E\Big(\sup_{\lambda_n\in\Omega_2} \hat{s}^{*b}(\Psi,\lambda_n,m)\Big) \text{ as } B\to\infty.$$

Note that $\sup_{\lambda_n\in\Omega_2} \hat{s}(\Psi,\lambda_n,m) \leq B^{-1}\sum_{b=1}^{B} \sup_{\lambda_n\in\Omega_2} \hat{s}^{*b}(\Psi,\lambda_n,m)$, it then follows immediately that $P(\sup_{\lambda_n\in\Omega_2} \hat{s}(\Psi,\lambda_n,m) \leq 1-\frac{p_0}{p}) \to 1$ and hence $P(\sup_{\lambda_n\in\Omega_2} \hat{s}(\Psi,\lambda_n,m) < 1-2\alpha_n) \to 1$. Therefore $P(\hat{\lambda}_n \in \Omega_2) \to 0$.

Finally, to show $P(\hat{\lambda}_n \in \Omega_3) \to 0$, it also suffices to show

$$P\Big(\sup_{\lambda_n\in\Omega_3} \hat{s}(\Psi,\lambda_n,m) < 1-2\alpha_n\Big) \to 1. \tag{14}$$

Assumption 2 implies that for any $j \in \mathcal{A}_T^c$ and some $j_1 \in \mathcal{A}_T^c$, when n is sufficiently large,

$$P(\cap_{\lambda_n\in\Omega_3}\{j \in \widehat{\mathcal{A}}_{\lambda_n}\}) \geq c_1(1/\varepsilon) > 0 \ \text{ and } \ P(\cap_{\lambda_n\in\Omega_3}\{j_1 \notin \widehat{\mathcal{A}}_{\lambda_n}\}) \geq c_2(\tau) > 0.$$

Therefore, it follows from the independence between two sub-samples that

$$\begin{aligned}
P\Big(\bigcap_{\lambda_n\in\Omega_3}\{\widehat{\mathcal{A}}^{*b}_{1\lambda_n} \neq \widehat{\mathcal{A}}^{*b}_{2\lambda_n}\}\Big) &\geq P\Big(\bigcap_{\lambda_n\in\Omega_3}\bigcup_{j\in\mathcal{A}_T^c}\{j \notin \widehat{\mathcal{A}}^{*b}_{1\lambda_n}, j \in \widehat{\mathcal{A}}^{*b}_{2\lambda_n}\}\Big) \\
&\geq P\Big(\bigcap_{\lambda_n\in\Omega_3}\{j_1 \notin \widehat{\mathcal{A}}^{*b}_{1\lambda_n}, j_1 \in \widehat{\mathcal{A}}^{*b}_{2\lambda_n}\}\Big) \\
&= P\Big(\bigcap_{\lambda_n\in\Omega_3}\{j_1 \notin \widehat{\mathcal{A}}^{*b}_{1\lambda_n}\}\Big)P\Big(\bigcap_{\lambda_n\in\Omega_3}\{j_1 \in \widehat{\mathcal{A}}^{*b}_{2\lambda_n}\}\Big), \\
&\geq c_1(1/\varepsilon)c_2(\tau).
\end{aligned}$$

Since the event $\bigcap_{\lambda_n\in\Omega_3}\{\widehat{\mathcal{A}}^{*b}_{1\lambda_n} \neq \widehat{\mathcal{A}}^{*b}_{2\lambda_n}\}$ implies that $\sup_{\lambda_n\in\Omega_3} \hat{s}^{*b}(\Psi,\lambda_n,m) \leq c_3$ with $c_3 = \max_{\mathcal{A}_1\neq\mathcal{A}_2} \kappa(\mathcal{A}_1,\mathcal{A}_2) \leq \frac{p-1}{p}$ where $\mathcal{A}_1,\mathcal{A}_2 \subset \{1,\cdots,p\}$, we have, for sufficiently large n,

$$P\Big(\sup_{\lambda_n\in\Omega_3} \hat{s}^{*b}(\Psi,\lambda_n,m) \leq c_3\Big) \geq c_1(1/\varepsilon)c_2(\tau).$$

Therefore, for sufficiently large n and any $b > 0$,

$$E\Big(\sup_{\lambda_n \in \Omega_3} \hat{s}^{*b}(\Psi, \lambda_n, m)\Big) \leq 1 - c_1(1/\varepsilon)c_2(\tau)(1-c_3).$$

Again, by the strong law of large number for U-statistics (Hoeffding, 1961) and the fact that $\sup_{\lambda_n \in \Omega_3} \hat{s}(\Psi, \lambda_n, m) \leq B^{-1} \sum_{b=1}^{B} \sup_{\lambda_n \in \Omega_3} \hat{s}^{*b}(\Psi, \lambda_n, m)$, we have

$$P\Big(\sup_{\lambda_n \in \Omega_3} \hat{s}(\Psi, \lambda_n, m) \leq 1 - c_1(1/\varepsilon)c_2(\tau)(1-c_3)\Big) \to 1.$$

For any ε, $c_1(1/\varepsilon)c_2(\tau)(1-c_3)$ is strictly positive and $\alpha_n \to 0$, we have

$$P\Big(\sup_{\lambda_n \in \Omega_3} \hat{s}(\Psi, \lambda_n, m) < 1 - 2\alpha_n\Big) \geq P\Big(\sup_{\lambda_n \in \Omega_3} \hat{s}(\Psi, \lambda_n, m) \leq 1 - c_1(1/\varepsilon)c_2(\tau)(1-c_3)\Big) \to 1,$$

and hence (14) is verified and $P\Big(\hat{\lambda}_n \in \Omega_3\Big) \to 0$.

Combining the above results, we have for any $\varepsilon > 0$,

$$\lim_{n\to\infty} \lim_{B\to\infty} P\Big(r_n/\varepsilon \leq \hat{\lambda}_n \leq \lambda_n^*\Big) = 1. \tag{15}$$

Furthermore, since for any $\varepsilon > 0$,

$$\begin{aligned} P(\widehat{\mathcal{A}}_{\hat{\lambda}_n} = \mathcal{A}_T) &\geq P\Big(\widehat{\mathcal{A}}_{\hat{\lambda}_n} = \mathcal{A}_T,\ r_n/\varepsilon \leq \hat{\lambda}_n \leq \lambda_n^*\Big) \\ &\geq P\Big(\bigcap_{r_n/\varepsilon \leq \lambda_n \leq \lambda_n^*} \{\widehat{\mathcal{A}}_{\lambda_n} = \mathcal{A}_T\}\Big) + P\Big(r_n/\varepsilon \leq \hat{\lambda}_n \leq \lambda_n^*\Big) - 1. \end{aligned}$$

Therefore, the desired selection consistency directly follows from (15) and Assumption 1 by letting $\varepsilon \to 0$. ■

Proof of Theorem 2: We prove Theorem 2 by similar approach as in the proof of Theorem 1. For any $\varepsilon > 0$, we denote $\Omega_1 = \{\lambda_n : \lambda_n > \lambda_n^*\}$, $\Omega_2 = \{\lambda_n : r_n^{-1}\lambda_n \leq \tau\}$ and $\Omega_3 = \{\lambda_n : \tau \leq r_n^{-1}\lambda_n \leq 1/\varepsilon\}$, where τ is selected so that $\tau < 1/\varepsilon$ and $p_n(1 - c_{1n}(\tau)) \succ \alpha_n$. Then we just need to show that $P(\hat{\lambda}_n \in \Omega_1 \cup \Omega_2 \cup \Omega_3) \to 0$. The probability $P(\hat{\lambda}_n \in \Omega_1) \to 0$ for any $\varepsilon > 0$ can be proved similarly as in Theorem 1.

In addition, Lemma 1 implies that $P(\hat{s}(\Psi, \lambda_n^*, m) \geq 1 - \alpha_n) \geq 1 - 2\varepsilon_n/\alpha_n$, and the definition of $\hat{\lambda}_n$ leads to $P(\hat{s}(\Psi, \hat{\lambda}_n, m) \geq (1-\alpha_n)(1-\alpha_n)) \geq 1 - 2\varepsilon_n/\alpha_n$, and hence

$$P\Big(\hat{s}(\Psi, \hat{\lambda}_n, m) \geq 1 - 2\alpha_n\Big) \geq P\Big(\hat{s}(\Psi, \hat{\lambda}_n, m) \geq (1-\alpha_n)(1-\alpha_n)\Big) \geq 1 - \frac{2\varepsilon_n}{\alpha_n} \to 1.$$

To show $P(\hat{\lambda}_n \in \Omega_2) \to 0$, it suffices to show $P(\sup_{\lambda_n \in \Omega_2} \hat{s}(\Psi, \lambda_n, m) < 1 - 2\alpha_n) \to 1$, which can be verified by slightly modifying the proof of (13). Assumption 2a implies that for any $j \in \mathcal{A}_T^c$ and $j_1 \in \mathcal{A}_T$, we have

$$P\Big(j \in \bigcap_{\lambda_n \in \Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) \geq c_{1n}(\tau) \text{ and } P\Big(j_1 \in \bigcap_{\lambda_n \in \Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) \geq 1 - \zeta_n.$$

As shown in Theorem 1, it implies that

$$P\Big(\{1,\ldots,p\} \in \bigcap_{\lambda_n \in \Omega_2} \widehat{\mathcal{A}}_{\lambda_n}\Big) \geq 1-(p_n-p_{0n})(1-c_{1n}(\tau))-p_{0n}\zeta_n,$$

and hence $E(\sup_{\lambda_n\in\Omega_2} \hat{s}^{*b}(\Psi,\lambda_n,m)) \leq 1-(p_n-p_{0n})(1-c_{1n}(\tau))-p_{0n}\zeta_n$. By the strong law of large number for U-statistics,

$$P\Big(\sup_{\lambda_n\in\Omega_2} \hat{s}(\Psi,\lambda_n,m) \leq 1-(p_n-p_{0n})(1-c_{1n}(\tau))-p_{0n}\zeta_n\Big) \to 1.$$

Therefore, $P(\sup_{\lambda_n\in\Omega_2} \hat{s}(\Psi,\lambda_n,m) < 1-2\alpha_n) \to 1$ provided that $p_n(1-c_{1n}(\tau)) \succ \alpha_n$ and $p_n\zeta_n \to 0$.

To show $P(\hat{\lambda}_n \in \Omega_3) \to 0$, it suffices to show

$$P\Big(\sup_{\lambda_n\in\Omega_3} \hat{s}(\Psi,\lambda_n,m) < 1-2\alpha_n\Big) \to 1. \tag{16}$$

Here (16) follows by modifying the proof of (14). According to $c_4 \leq (p_n-1)/p_n$, we have

$$E\Big(\sup_{\lambda_n\in\Omega_3} \hat{s}^{*b}(\Psi,\lambda_n,m)\Big) \leq 1-p_n^{-1}c_{1n}(1/\varepsilon)c_{2n}(\tau).$$

Therefore, following the same derivation as in Theorem 1, we have

$$P\Big(\sup_{\lambda_n\in\Omega_3} \hat{s}(\Psi,\lambda_n,m) \leq 1-p_n^{-1}c_{1n}(1/\varepsilon)c_{2n}(\tau)\Big) \to 1.$$

This, together with the assumptions that $\alpha_n \prec p_n^{-1}c_{1n}(1/\varepsilon)c_{2n}(\tau)$ for any ε and τ, leads to the convergence in (16), which completes the proof of Theorem 2. ■

References

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19, 716-723, 1974.

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

P. Breheny and J. Huang. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Annals of Applied Statistics*, 5, 232-253, 2011.

J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20, 37-46, 1960.

P. Craven and G. Wahba. Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerische Mathematik*, 31, 317-403, 1979.

B. Efron, T. Hastie, I. Johnstone and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32, 407-451, 2004.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96, 1348-1360, 2001.

J. Fan and R. Li. Statistical Challenges with high dimensionality: Feature selection in knowledge discovery. In *Proceedings of the International Congress of Mathematicians*, 3, 595-622, 2006.

J. Fan and H. Peng. Nonconcave penalized likelihood with a diverging number of parameters. *Annals of Statistics*, 32, 928-961, 2004.

W. Hoeffding. The strong law of large numbers for U-statistics. *Institute of Statistical Mimeograph Series, No. 302, University of North Carolina*, 1961.

J. Huang, S. Ma and C.H. Zhang. Adaptive Lasso for sparse high-dimensional regression models. *Statistica Sinica*, 18, 1603-1618, 2008.

J. Huang and H. Xie. Asymptotic oracle properties of SCAD-penalized least squares estimators. *IMS Lecture Notes - Monograph Series, Asymptotics: Particles, Processes and Inverse Problems*, 55, 149-166, 2007.

C. Mallows. Some comments on Cp. *Technometrics*, 15, 661-675, 1973.

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society, Series B*, 72, 414-473, 2010.

G.E. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6, 461-464, 1978.

X. Shen, W. Pan and Y. Zhu. Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107, 223-232, 2012.

T.A. Stamey, J.N. Kabalin, J.E. McNeal, I.M. Johnstone, F. Freiha, E.A. Redwine and N. Yang. Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate: II. Radical prostatectomy treated patients. *Journal of Urology*, 141, 1076-1083, 1989.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, Series B*, 36, 111-147, 1974.

W. Sun, J. Wang and Y. Fang. Regularized K-means clustering of high-dimensional data and its asymptotic consistency, *Electronic Journal of Statistics*, 6, 148-167, 2012.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58, 267-288, 1996.

H. Wang and C. Leng. A note on adaptive group Lasso. *Computational Statistics and Data Analysis*, 52, 5277-5286, 2008.

H. Wang, R. Li and C.L. Tsai. Tuning parameter selectors for the smoothly clipped absolute deviation method. *Biometrika*, 94, 553-568, 2007.

H. Wang, B. Li and C. Leng. Shrinkage tuning parameter selection with a diverging number of parameters. *Journal of the Royal Statistical Society, Series B*, 71, 671-683, 2009.

J. Wang. Consistent selection of the number of clusters via cross validation. *Biometrika*, 97, 893-904, 2010.

L. Xue, A. Qu and J. Zhou. Consistent model selection for marginal generalized additive model for correlated data. *Journal of the American Statistical Association*, 105, 1518-1530, 2010.

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.

Y. Zhang, R. Li and C.L. Tsai. Regularization parameter selections via generalized information criterion. *Journal of the American Statistical Association*, 105, 312-323, 2010.

P. Zhao and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7, 2541-2563, 2006.

H. Zou. The adaptive Lasso and its oracle properties. *Journal of the American Statistical Association*, 101, 1418-1429, 2006.

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

H. Zou, T. Hastie and R. Tibshirani. On the "Degree of Freedom" of the Lasso. *Annals of Statistics*, 35, 2173-2192, 2007.

H. Zou and H. Zhang. On the adaptive elastic-net with a diverging number of parameters. *Annals of Statistics*, 37, 1733-1751, 2009.

Journal of Machine Learning Research 14 (2013) 3441-3492 Submitted 6/11; Revised 7/13; Published 11/13

Learning Theory Analysis for Association Rules and Sequential Event Prediction

Cynthia Rudin RUDIN@MIT.EDU
Sloan School of Management
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139, USA

Benjamin Letham BLETHAM@MIT.EDU
Operations Research Center
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139, USA

David Madigan MADIGAN@STAT.COLUMBIA.EDU
Department of Statistics
Columbia University
1255 Amsterdam Avenue
New York, NY 10027, USA

Editor: John Shawe-Taylor

Abstract

We present a theoretical analysis for prediction algorithms based on association rules. As part of this analysis, we introduce a problem for which rules are particularly natural, called "sequential event prediction." In sequential event prediction, events in a sequence are revealed one by one, and the goal is to determine which event will next be revealed. The training set is a collection of past sequences of events. An example application is to predict which item will next be placed into a customer's online shopping cart, given his/her past purchases. In the context of this problem, algorithms based on association rules have distinct advantages over classical statistical and machine learning methods: they look at correlations based on subsets of co-occurring past events (items a and b imply item c), they can be applied to the sequential event prediction problem in a natural way, they can potentially handle the "cold start" problem where the training set is small, and they yield interpretable predictions. In this work, we present two algorithms that incorporate association rules. These algorithms can be used both for sequential event prediction and for supervised classification, and they are simple enough that they can possibly be understood by users, customers, patients, managers, etc. We provide generalization guarantees on these algorithms based on algorithmic stability analysis from statistical learning theory. We include a discussion of the strict minimum support threshold often used in association rule mining, and introduce an "adjusted confidence" measure that provides a weaker minimum support condition that has advantages over the strict minimum support. The paper brings together ideas from statistical learning theory, association rule mining and Bayesian analysis.

Keywords: statistical learning theory, algorithmic stability, association rules, sequence prediction, associative classification

1. Introduction

Consider the problem of predicting the next event within a current event sequence, given a "sequence database" of past event sequences to learn from. We might wish to do this, for instance, using data generated by a customer placing items into the virtual basket of an online grocery store such as NYC's Fresh Direct, Peapod by Stop & Shop, or Roche Bros. The customer adds items one by one into the current basket, creating a sequence of events. The customer has identified him- or herself, so that all past orders are known. After each item selection, a confirmation screen contains a small list of recommendations for items that are not already in the basket. If the store can find patterns within the customer's past purchases, it may be able to accurately recommend the next item that the customer will add to the basket. Another example is to predict each next symptom of a sick patient, given the patient's past sequence of symptoms and treatments, and a database of the timelines of symptoms and treatments for other patients. We call the problem of predicting these sequentially revealed events based on past sequences of events "sequential event prediction."

In these examples, a subset of past events (for instance, a set of ingredients for a particular recipe) can be useful in predicting the next event. In order to make predictions using subsets of past events, we employ *association rules* (Agrawal et al., 1993). An association rule in this setting is an implication $a \rightarrow b$ (such as *lettuce and carrots* $\rightarrow$ *tomatoes*), where a is a subset of items, and b is a single item. The association rule approach has the distinct advantage in being able to directly model underlying conditional probabilities $P(b|a)$ eschewing the linearity assumptions underlying many classical supervised classification, regression, and ranking methods. Rules also yield predictive models that are interpretable, meaning that for the rule $a \rightarrow b$, it is clear that b was recommended because a is satisfied.

The association rules approach makes predictions from subsets of co-occurring past events. Using subsets may make the estimation problem much easier, because it helps avoid problems with the curse of dimensionality. For instance $P(\textit{tomatoes} \mid \textit{lettuce and carrots})$ could be much easier to estimate than $P(\textit{tomatoes} \mid \textit{lettuce}, \textit{carrots}, \textit{pears}, \textit{potatoes}, \textit{ketchup}, \textit{eggs}, \textit{bread}, \text{etc.})$. This is precisely why learning algorithms created from rules can be helpful for the "cold start" problem in recommender systems, where predictions need to be made when there are not enough data available to accurately compute the full probability of a new item being purchased.

There are two main contributions in this work: a generalization analysis for association-rule-based algorithms, and a formal definition of the problem of sequential event prediction. An important part of the rule-based analysis is how a fundamental property of a rule, namely the "support," is incorporated into the generalization bounds. The "support" of an itemset is the number of times that the itemset has appeared in the sequence database. For instance, the support of *lettuce* is the number of times lettuce has been purchased in the past. Typically in association rule mining, a strict minimum support threshold condition is placed on the support of itemsets within a rule, so that rules falling below the minimum support threshold are simply discarded. The idea of a condition on the support is not shared with other types of supervised learning algorithms, since they do not use subsets in the same way as when using rules. Thus a new aspect of generalization is explored in this analysis in that it handles predictions created from subsets of data. In classical supervised learning paradigms, bounds scale only with the sample size, and a large sample is necessary to create a generalization guarantee. In the context of association rules, the minimum support threshold forces predictions to be made only when there are enough data. Thus, in the association rules analysis, there are now two mechanisms for generalization: first a large sample, and second,

a minimum support. These are separate mechanisms, in the sense that it is possible to generalize with a somewhat small sample size and a large minimum support threshold, and it is also possible to generalize with a large sample size and no support threshold. We thus derive two types of bounds: large sample bounds, which scale with the sample size, and small sample bounds, which scale with the minimum support of rules. Using both large and small sample bounds (that is, the minimum of the two bounds) gives a complete picture. The large sample bounds are of order $O(\sqrt{1/m})$ as in classical analysis of supervised learning, where m denotes the number of event sequences in the database, that is, the number of past baskets ordered by the online grocery store customer.

Most of our bounds are derived using a specific notion of algorithmic stability called "pointwise hypothesis stability." The original notions of algorithmic stability were invented in the 1970's and have been revitalized recently (Devroye and Wagner, 1979; Bousquet and Elisseeff, 2002), the main idea being that algorithms may be better able to generalize if they are insensitive to small changes in the training data such as the removal or change of one training example. The pointwise hypothesis stability specifically considers the average change in loss that will occur at one of the training examples if that example is removed from the training set. Our generalization analysis uses conditions on the minimum support of rules in order to bound the pointwise hypothesis stability.

There are two algorithms considered in this work. At the core of each algorithm is a method for rank-ordering association rules where the list of possible rules is generated using the customer's past purchase history and subsets of items within the current basket. These algorithms build off of the rule mining literature that has been developing since the early 1990's (Agrawal et al., 1993) by using an application-specific rule mining method as a subroutine. Our algorithms are interpretable in two different ways: the predictive model coming out of the algorithm is interpretable, and the whole algorithm for producing the predictive model is interpretable. In other words, the algorithms are straightforward enough that they can be understood by users, customers, patients, managers, etc. Further, the rules within the predictive model can provide a simple reason to the customer why an item might be relevant, or identify that a key ingredient is missing from a particular recipe. The rules provide "IF,THEN,ELSE" conditions, and yield models of the same form as those from the expert systems literature from the early days of artificial intelligence (Jackson, 1998). Many authors have emphasized the importance of interpretability and explanation in predictive modeling (see, for example, the work of Madigan et al., 1997).

The first of the two algorithms considered in this work uses a fixed minimum support threshold to exclude rules whose itemsets occur rarely. Then the remaining rules are ranked according to the "confidence," which for rule $a \rightarrow b$ is the empirical probability that b will be in the basket given that a is in the basket. The right-hand sides of the highest ranked rules will be recommended by the algorithm. However, the use of a strict minimum support threshold is problematic for several well-known reasons, for instance it is known that important rules ("nuggets," which are rare but strong rules) are often excluded by a minimum support threshold condition.

The other algorithm introduced in this work provides an alternative to the minimum support threshold, in that rules are ranked by an "adjusted" confidence, which is a simple Bayesian shrinkage estimator of the probability of a rule $P(b|a)$. The right-hand sides of rules with the highest adjusted confidence are recommended by the algorithm. For this algorithm, the generalization guarantee is smoothly controlled by a parameter K, which provides only a weak (less restrictive) minimum support condition. The key benefits of an algorithm based on the adjusted confidence are that: 1) it allows the possibility of choosing very accurate (high confidence) rules that have appeared very few times in the training set (low support), and 2) given two rules with the same or similar prediction

accuracy on the training set (confidence), the rule that appears more frequently (higher support) achieves a higher adjusted confidence and is thus preferred over the other rule.

All of the bounds are tied to the measure of quality (the loss function) used within the analysis. We would like to directly compare the performance of algorithms for various settings of the adjusted confidence's K parameter (and for the minimum support threshold θ). It is problematic to have the loss defined using the same K value as the algorithm, in that case we would be using a different method of evaluation for each setting of K, and we would not be able to directly compare performance across different settings of K. To allow a direct comparison, we select one reference value of the adjusted confidence, called K_r (r for "reference"), and the loss depends on K_r rather than on K. The bounds are written generally in terms of K_r. The special case $K_r = 0$ is where the algorithm is evaluated with respect to the confidence measure. The small sample bounds for the adjusted confidence algorithm have two terms: one that generally decreases with K (as the support increases, there is better generalization) and the other that decreases as K gets closer to K_r (better generalization as the algorithm is closer to the way it is being measured). These two terms are thus agreeing if $K_r > K$ and competing if $K_r < K$. In practice, the choice of K can be determined in several ways: K can be manually determined (for instance by the customer), it can be set using side information as considered by McCormick et al. (2012), or it can be set via cross-validation on an extra hold-out set.

The novel elements of the paper include: 1) generalization analysis that incorporates the use of association rules, for both classification and sequential event prediction, 2) the algorithm based on adjusted confidence, where the adjusted confidence is a Bayesian version of the confidence, 3) the definition of a new supervised learning problem, namely sequential event prediction. The work falls in the intersection of several fields that are rarely connected: association rule mining and associative classification, supervised machine learning and generalization bounds from statistical learning theory, and Bayesian analysis.

In terms of applications, the definition of "sequential event prediction" was inspired by, but not restricted to, online grocery stores. Examples are Fresh Direct, Amazon.com grocery, and netgrocer.com. Many supermarket chains with local outlets also offer an online shop-and-delivery option, such as Peapod (paired with Stop & Shop and Giant). Other online retailers and recommendation engines may benefit from ranking algorithms that are transparent to the user like amazon.com's "customers who purchased this also purchased that" recommender system. The same techniques used to solve the sequential event prediction problem could be used in medical applications to predict, for instance, the winners at each round of a tournament (e.g, the winners of games in a football season), or the next move of a video game player in order to design a more interesting game. The work of McCormick et al. (2012) contains a Bayesian algorithm, based on the analysis introduced in this paper, for predicting conditions of medical patients in a clinical trial. The work of Letham et al. (2013b) uses empirical risk minimization to solve sequential event prediction problems dealing with email recipient recommendation, healthcare, and cooking.

Section 2 describes the two rule-based prediction algorithms, one based on a hard thresholding of the support (min support) and the other based on the soft thresholding (adjusted confidence). Section 3 formally defines sequential event prediction. Section 4 provides the generalization analysis, Section 5 contains proofs, and Section 6 provides experimental validation. Section 7 contains a summary of relevant literature. Appendix A discusses the suitability of regression approaches for solving the sequential event prediction problem. Appendix B provides additional experimental results. Appendix C contains an additional proof.

2. Derivation of Algorithms

We assume an interface similar to that of Fresh Direct, where users add items one by one into the basket. After each selection, a confirmation screen contains a handful of recommendations for items that are not already in the customer's basket. The customer's past orders are known.

The set of items is $\mathcal{X}$, for instance $\mathcal{X}$=$\{$*apples, bananas, pears*, etc$\}$; $\mathcal{X}$ is the set of possible events. The customer has a past history of orders S which is a collection of m baskets, $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$; S is the sequence database. The customer's current basket is usually denoted by $B \subset \mathcal{X}$; B is the current sequence. An algorithm uses B and S to find rules $a \to b$, where a is in the basket and b is not in the basket. For instance, if *salsa* and *guacamole* are in the basket B and also if *salsa, guacamole* and *tortilla chips* were often purchased together in S, then the rule (*salsa* and *guacamole*) $\to$ *tortilla chips* might be used to recommend *tortilla chips*.

The support of a, written $\mathrm{Sup}(a)$ or $\#a$, is the number of times in the past the customer has ordered itemset a,

$$\mathrm{Sup}(a) := \#a := \sum_{i=1}^{m} \mathbb{1}_{[a \subseteq z_i]}.$$

If $a = \varnothing$, meaning a contains no items, then $\#a := \sum_i 1 = m$. The confidence of a rule $a \to b$ is denoted "Conf" or "$f_{S,0}$":

$$\mathrm{Conf}(a \to b) := f_{S,0}(a,b) := \frac{\#(a \cup b)}{\#a},$$

the fraction of times b is purchased given that a is purchased. It is an estimate of the conditional probability of b given a. Ultimately an algorithm should order rules by conditional probability; however, the rules that possess the highest confidence values often have a left-hand side with small support, and their confidence values do not yield good estimates for the true conditional probabilities. Note that $a \cup b$ is the union of the set a with item b (the intersection is empty). In this work we introduce the "adjusted" confidence as a remedy for this problem: The *adjusted confidence* for rule $a \to b$ is:

$$f_{S,K}(a,b) \quad := \frac{\#(a \cup b)}{\#a + K}.$$

The adjusted confidence for $K = 0$ is equivalent to the confidence.

The adjusted confidence is a particular Bayesian estimate of the confidence. Specifically, assuming a beta prior distribution for the confidence, the posterior mean is given by:

$$\hat{p} = \frac{L + \#(a \cup b)}{L + K + \#a},$$

where L and K denote the parameters of the beta prior distribution. The beta distribution is the "conjugate" prior distribution for a binomial likelihood. For the adjusted confidence we choose $L = 0$. This choice yields the benefits of the lower bounds derived in the remainder of this section, and the stability properties described later. The prior for the adjusted confidence tends to bias rules towards the *bottom* of the ranked list. Any rule achieving a high adjusted confidence must overcome this bias.

Other possible choices for L and K are meaningful. For instance we could choose the following:

- Collaborative filtering prior: have $L/(L+K)$ represent the probability of purchasing item b given that item a was purchased, calculated over a subset of other customers. This biases estimates of the target user's behavior towards the "average" user.
- Revenue management prior: choose L and K based on the item's price, so more expensive items are more likely to be recommended.
- Time dependent prior: use only the customer's most recent orders, and choose L and K to summarize the user's behavior before this point.

A rule cannot have a high adjusted confidence unless it has a large enough confidence and also a large enough support on the left-hand side. To see this, consider the case when we take $f_{S,K}(a,b)$ large, meaning for some η, we have $f_{S,K}(a,b) > \eta$, implying:

$$\text{Conf}(a \to b) = f_{S,0}(a,b) > \eta \frac{\#a+K}{\#a} \geq \eta,$$

$$\text{Sup}(a) = \#a \geq (\#a+K)\left[\frac{\#(a \cup b)}{\#a+K}\right] > (\#a+K)\eta, \text{ implying } \text{Sup}(a) = \#a > \frac{\eta K}{1-\eta}. \quad (1)$$

And further, expression (1) implies:

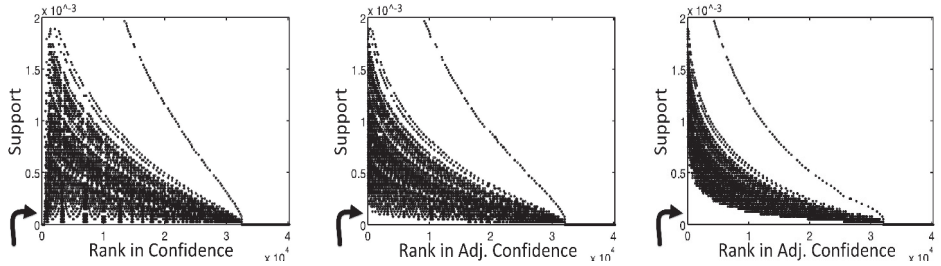
$$\text{Sup}(a \cup b) = \#(a \cup b) > \eta(\#a+K) > \eta K/(1-\eta).$$

Thus, rules attaining high values of adjusted confidence have a lower bound on confidence, and a lower bound on support of both the right and left-hand sides, which means a better estimate of the conditional probability. The bounds clearly do not provide any advantage when $K = 0$ and the confidence is used.

As K increases, rules with low support are heavily penalized, so they tend not to be at the top of the list. On the other hand, such rules might be chosen when all other rules have low confidence. That is an advantage of having no firm minimum support cutoff: "nuggets" that have fairly low support may filter to the top. Figure 1 illustrates this by showing the support of rules ordered by adjusted confidence, for two values of K, using a transactional data set "T25I10D10KN200" from the IBM Quest Market-Basket Synthetic Data Generator (Agrawal and Srikant, 1994) which mimics a retail data set.[1] We use all rules with either one or no items on the left and one item on the right (as produced for instance by *GenRules*, presented in Algorithm 1). On each scatter plot, each of the rules is represented by a point. The rules are ordered on the x-axis by adjusted confidence, and the support of the rule is indicated on the y-axis. As K increases, rules with the highest adjusted confidence are required to achieve a higher support, as can be seen from the gap in the lower left corner of the scatter plot for larger K.

We now formally state the recommendation algorithms. Both algorithms use a subroutine for mining association rules to generate a set of candidate rules. *GenRules* (Algorithm 1) is one of the simplest such rule mining algorithms, which in practice should be replaced by a rule mining algorithm that retrieves rules tailored to the application. There is a vast literature on such algorithms since the field of association rule mining evolved on their development, *e.g.* Apriori (Agrawal et al., 1993). *GenRules* requires a set A which is the set of allowed left-hand sides of rules.

1. The data set generated is T25I10D10KN200 that contains 10K transactions, 200 items, and where the average length of transactions is 25 and the average pattern length is 10.



$K = 0$ $K = 10$ $K = 50$

Figure 1: Support vs. rank in adjusted confidence for $K = 0, 10, 50$. Rules with the highest adjusted confidence are on the left.

Algorithm 1: *Subroutine GenRules*.

Input: $(S, B, \mathcal{X})$, that is, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, set of items $\mathcal{X}$

Output: Set of all rules $\{a_j \to b_j\}_j$ where b_j is a single item that is not in the basket B, and where a_j is either a subset of items in the basket B, or else it is the empty set. Also the left-hand side a_j must be allowed (meaning it is in A). That is, output rules $\{a_j \to b_j\}_j$ such that $b_j \in \mathcal{X} \backslash B$ and $a_j \subseteq B \subset \mathcal{X}$ with $a_j \in A$, or $a_j = \varnothing$.

2.1 Max Confidence, Min Support Algorithm

The max confidence, min support algorithm, shown as Algorithm 2, is based on the idea of eliminating rules whose itemsets occur rarely, which is commonly done in the rule-mining literature. For this algorithm, the rules are ranked by confidence, and rules that do not achieve a predetermined fixed minimum support threshold are completely omitted. The algorithm recommends the right-hand sides from the top ranked rules. Specifically, if c items are to be recommended to the user, the algorithm picks the top ranked c distinct items.

It is common that the minimum support threshold is imposed on the right and left side $\mathrm{Sup}(a \cup b) \geq \theta$; however, as long as $\mathrm{Sup}(a)$ is large, we can get a reasonable estimate of $P(b|a)$. In that sense, it is sufficient (and less restrictive) to impose the minimum support threshold on the left side: $\mathrm{Sup}(a) \geq \theta$. Here θ is a number determined beforehand (for instance, the support of the left must be at least 5 items). In this work, we only have a required minimum support on the left side. As a technical note, we might worry about the minimum support threshold being so high that there are no rules that meet the threshold. This is actually not a major concern because of the minimum support being imposed only on the left-hand side: as long as $m \geq \theta$, all rules $\varnothing \to b$ meet the minimum support threshold.

The thresholded confidence is denoted by $\bar{f}_{S,\theta}$:

$$\bar{f}_{S,\theta}(a,b) := f_{S,0}(a,b) \text{ if } \#a \geq \theta, \text{and } \bar{f}_{S,\theta}(a,b) := 0 \text{ otherwise.}$$

Algorithm 2: Max Confidence, Min Support Algorithm.

Input: $(\theta, \mathcal{X}, S, B, \textit{GenRules}, c)$, that is, minimum threshold parameter θ, set of items $\mathcal{X}$, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, *GenRules* generates candidate rules $\textit{GenRules}(S, B, \mathcal{X}) = \{a_j \to b_j\}_j$, number of recommendations $c \geq 1$

Output: Recommendation List, which is a subset of c items in $\mathcal{X}$

1 Apply $\textit{GenRules}(S, B, \mathcal{X})$ to get rules $\{a_j \to b_j\}_j$ where a_j is in the basket B and b_j is not.

2 Compute score for each rule $a_j \to b_j$ as $\bar{f}_{S,\theta}(a_j, b_j) = f_{S,0}(a_j, b_j) = \frac{\#(a_j \cup b_j)}{\#a_j}$ when support $\#a_j \geq \theta$, and $\bar{f}_{S,\theta}(a_j, b_j) = 0$ otherwise.

3 Reorder rules by decreasing score.

4 Find the top c rules with distinct right-hand sides, and let Recommendation List be the right-hand sides of these rules.

Algorithm 3: Adjusted Confidence Algorithm.

Input: $(K, \mathcal{X}, S, B, \textit{GenRules}, c)$, that is, parameter K, set of items $\mathcal{X}$, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, *GenRules* generates candidate rules $\textit{GenRules}(S, B, \mathcal{X}) = \{a_j \to b_j\}_j$, number of recommendations $c \geq 1$

Output: Recommendation List, which is a subset of c items in $\mathcal{X}$

1 Apply $\textit{GenRules}(S, B, \mathcal{X})$ to get rules $\{a_j \to b_j\}_j$ where a_j is in the basket B and b_j is not.

2 Compute adjusted confidence of each rule $a_j \to b_j$ as $f_{S,K}(a_j, b_j) = \frac{\#(a_j \cup b_j)}{\#a_j + K}$.

3 Reorder rules by decreasing adjusted confidence.

4 Find the top c rules with distinct right-hand sides, and let Recommendation List be the right-hand sides of these rules.

2.2 Adjusted Confidence Algorithm

The adjusted confidence algorithm is shown as Algorithm 3. A chosen value of K is used to compute the adjusted confidence for each rule, and rules are then ranked according to adjusted confidence.

The definition of the adjusted confidence makes an implicit assumption that the order in which items were placed into previous baskets is irrelevant. It is easy to include a dependence on the order by defining a "directed" version of the adjusted confidence, and calculations can be adapted accordingly. The numerator of the adjusted confidence becomes the number of past orders where a is placed in the basket *before* b.

$$f_{S,K}^{(\text{directed})}(a, b) \quad = \quad \frac{\#\{(a \cup b) : b \text{ follows } a\}}{\#a + K}.$$

2.3 Rule Selection

In classical supervised machine learning problems, like classification and regression, designing features is one of the main engineering challenges. In association rule modeling, the analogous challenge is designing the allowed sets of items for the left and right sides of rules. For instance, we can choose to capture only positive correlations, as if customers were purchasing items from several independent recipes. The present work considers mainly positive correlations, for the purpose of exposition and to keep things simple. Beyond this, it is easily possible to capture negative corre-

lations between items by creating "negation" items, such as $\neg b$. As an example of using negation rules in the ice cream category, we impose that for *vanilla* to be on the right, both *chocolate* and *strawberry* need to be on the left, in either their usual form or negated. Of these, the rule that is used corresponds to the current basket. In that case, $\neg$*chocolate*, $\neg$*strawberry* $\rightarrow$ *vanilla* could have a high score in order to recommend *vanilla* when *chocolate* and *strawberry* are not in the basket, whereas *chocolate*, $\neg$*strawberry* $\rightarrow$ *vanilla* might have a low score, conveying that since *chocolate* is already in the basket that *vanilla* should not be recommended. Alternatively, we could create a negation item $\neg$*ice_cream* indicating that the basket contains no ice cream presently, so *sprinkles* + $\neg$*ice_cream* $\rightarrow$ *vanilla* could have a high score.

We can also use negation items on the right, where if there is a rule $a \rightarrow \neg b$ that receives a higher score (confidence or adjusted confidence) than any other rules recommending b, we can choose not to recommend b. Rules can be designed to capture higher level correlations in specific regimes, for instance the allowed set A can contain up to three items in one product category, but only two items in another. It is not practical in general to exhaustively enumerate and use all possible rules in a rule modeling algorithm due to problems with computational complexity. The key is to find a small but good set of rules, for instance the set of rules containing exhaustively all subsets of 1, 2, or 3 items on the left; or perhaps use the top rules that come out of the Apriori algorithm (Agrawal et al., 1993). In Section 7 we provide citations to surveys on association rule mining and associative classification that discuss this important issue of rule-construction and rule-engineering.

2.4 Modeling Assumption

The general modeling assumption that we make with the two algorithms above can be written as follows, where current basket B is composed of items $b_1, \ldots b_t$, and X_i is the random variable governing whether item i will be placed into the basket next:

$$\begin{aligned} &\operatorname*{argmax}_{\substack{i=1,\ldots,m \\ i \notin B}} P(X_i = 1 | X_{b_1} = 1, X_{b_2} = 1, \ldots, X_{b_t} = 1) \\ &= \operatorname*{argmax}_{\substack{i=1,\ldots,m \\ i \notin B}} \max_{\substack{a \in A \\ a \subseteq \{b_1,\ldots,b_t\}}} P(X_i = 1 | X_{a_1} = 1, X_{a_2} = 1, \ldots). \end{aligned}$$

This expression states that the most likely item to be added next into the basket can be identified using a subset of items in the basket, denoted a. That subset is restricted to fall into a class A which is chosen based on the application at hand and the ease in which that subset can be searched. The set A determines the hypothesis space for learning, and it would be chosen differently as we move from the small sample regime to the large sample regime, so that the right side of this expression would eventually look just like the left side when the sample is large.

The choice of A can help with the problem of "curse of dimensionality" by allowing us to look at small subsets on the left. A similar example to the one in the introduction is P(machine will break | a particular part is old) could be much easier to estimate accurately than the full probability P(machine will break | part 1 did poorly at last inspection, part 2 is very old, part 3 is new, part 4 is ok,..., part 612 is ok, etc.). The large dimensionality would likely be a problem when estimating the full probability. Further, the approximation also could actually be sufficient to estimate the full probability. We note that there are circumstances in which it is natural to only consider positive correlations. In the example of equipment failure, for instance, individual component failures would always increase the risk of overall failure. More typically, however, consideration of both positive and negative correlations will be important.

Our modeling assumption aligns with sequential event prediction, where only part of a sequence is available to make a prediction at time t. This is a case where standard linear modeling approaches do not naturally apply, since one would need to make a linear combination of terms, some of which are unrealized. We discuss this more in Appendix A.

3. Definition of Sequential Event Prediction

For simplicity in notation, at each time the algorithm recommends only one item, $c = 1$. A basket z consists of an ordered (permuted) set of items, $z \in 2^{\mathcal{X}} \times \Pi$, where $2^{\mathcal{X}}$ is the set of all subsets of $\mathcal{X}$, and Π is the set of permutations over at most $|\mathcal{X}|$ elements. We have a training set of m baskets $S = \{z_i\}_{1\ldots m}$ that are the customer's past orders. Denote $z \sim \mathcal{D}$ to mean that basket z is drawn randomly (iid) according to distribution $\mathcal{D}$ over the space of possible items in baskets and permutations over those items, $2^{\mathcal{X}} \times \Pi$. The t^{th} item added to the basket is written $z_{\cdot,t}$, where the dot is just a placeholder for the generic basket z. The t^{th} element of the i^{th} basket in the training set is written $z_{i,t}$. We define the number of items in basket z by T_z, that is, $T_z := |z|$. We introduce a generic scoring function $f_S : (a,b) \mapsto \mathbb{R}$ where a is a subset of items and b is a single item. The input a to the score is $\{z_{\cdot,1},\ldots,z_{\cdot,t}\}$ or is a subset of $\{z_{\cdot,1},\ldots,z_{\cdot,t}\}$. For now we let a be the full set $\{z_{\cdot,1},\ldots,z_{\cdot,t}\}$. The input b is an item that is not already in the basket, $b \in \mathcal{X}\backslash\{z_{\cdot,1},\ldots,z_{\cdot,t}\}$. The scoring function f_S comes from an algorithm that takes data set S as input. We can consider f_S to be parameterized, and the algorithm will learn the parameters of f_S from S.

If the score $f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b)$ is larger than that of $f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},z_{\cdot,t+1})$, it means that the algorithm recommended the wrong item. The loss function below counts the proportion of times this happens for each basket.

$$\ell_{0-1}(f_S,z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1}\begin{cases} 1 & \text{if } f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},z_{\cdot,t+1}) - \max_{b\in\mathcal{X}\backslash\{z_{\cdot,1},\ldots,z_{\cdot,t}\}} f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b) \le 0 \\ 0 & \text{otherwise.}\end{cases}$$

(Note that if z contains all items in $\mathcal{X}$, then the recommendation for the last item is deterministic, so we would not count it towards the loss.) The true error for sequential event prediction is an expectation of the loss with respect to $\mathcal{D}$, and is again a random variable since the training set S is random.

$$\text{TrueErr}(f_S) := \mathbb{E}_{z\sim\mathcal{D}}\ell_{0-1}(f_S,z).$$

The empirical risk is the average loss with respect to S:

$$\text{EmpErr}(f_S) := \frac{1}{m}\sum_{i=1}^{m}\ell_{0-1}(f_S,z_i).$$

The loss is bounded (by 1), the baskets are chosen independently, and the empirical risk is an average of iid random variables and the true risk is the expectation. Thus, the problem fits into the traditional scope of statistical learning, and the loss can be used within concentration arguments to obtain generalization bounds.

In the analysis below, we build the full algorithm for constructing f_S into the notation. The algorithms above are simple enough that they can be encoded within the same line of notation. To do this we will say that f_S acts on the the subset of $\{z_{\cdot,1},\ldots,z_{\cdot,t}\}$ within $\mathcal{A}$ that has the maximum

score. For instance, if we are using the adjusted confidence algorithm,

$$f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b) := \max_{a\in A, a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\}} f_{S,K}(a,b).$$

The 0-1 loss is not smooth, so we will often use a smooth convex upper bound for the loss within the bounds. Specifically, for the way we have defined sequential event prediction, if any item has a higher score than the next item added, the algorithm incurs an error. (Even if that item is added later on, the algorithm incurs an error at this timestep.) To measure the size of that error, we can use the 0-1 loss, indicating whether or not our algorithm gave the highest score to the next item added. However, the 0-1 loss does not capture how close our algorithm was to correctly predicting the next item, though this information might be useful in determining how well the algorithm will generalize. We approximate the 0-1 loss using a modified loss that decays linearly near the discontinuity. This modified loss allows us to consider differences in adjusted confidence, not just whether one is larger than another:

$$\begin{aligned}&|(\text{adjusted conf. of highest-scoring-correct rule})\\&\quad-(\text{adjusted conf. of highest-scoring-incorrect rule})|.\end{aligned}$$

However, as discussed in the introduction, if we adjust the loss function's K value to match the adjusted confidence K value, then we cannot fairly compare the algorithm's performance using two different values of K. An illustration of this point is that for large K, all adjusted confidence values are $\ll 1$, and for small K, the adjusted confidence can be ≈ 1; differences in adjusted confidence for small K cannot be directly compared to those for large K. Since we want to directly compare performance as K is adjusted, we fix an evaluation measure that is separate from the choice of K. Specifically, we use the difference in adjusted confidence values with respect to a reference K_r:

$$\begin{aligned}&|(\{\text{adjusted conf.}\}_{K_r}\text{of highest-scoring-correct rule}_K)\\&\quad-(\{\text{adjusted conf.}\}_{K_r}\text{of highest-scoring-incorrect rule}_K)|.\end{aligned}\tag{2}$$

The reference K_r is a parameter of the loss function, whereas K is a parameter of an algorithm. We set $K_r = 0$ to measure loss using the difference in confidence, and $K = 0$ for an algorithm that chooses rules according to the confidence. As K gets farther from K_r, the algorithm is more distant from the way it is being evaluated, which leads to worse generalization. Note that for $K_r = K$, the 0-1 loss is the same as the sign of (2).

A similar loss will be used in classification, where we incur an error if the adjusted confidence of the incorrect label is higher than that of the correct label.

4. Generalization

Our goal in this section is to provide a foundation for supervised learning with association rules, and also a foundation for sequential event prediction. We will consider several quantities that may be important in the learning process: m, K or θ, the size of the set of possible itemsets $|A|$, and the probability of the least probable itemsets and items.

As part of this section, we establish bounds for vanilla supervised binary classification with rules. Specifically we consider "max-score" association rule classifiers. For a given example, a max-score classifier assigns a score to the label +1 and a score to the label -1, and chooses the label

corresponding to the higher of the two scores. Max-score association rule classifiers are a special type of "associative classifier" (Liu et al., 1998) and are also a type of "decision list" (Rivest, 1987). The result in 4.2 is a uniform bound based on the VC dimension of the set of max-score classifiers. This bound does not depend explicitly on K, which we hypothesize is an important quantity for the learning process.

In order to understand how K might affect learning, we use algorithmic stability analysis. This approach originated in the 1970's (Rogers and Wagner, 1978; Devroye and Wagner, 1979) and was revitalized by Bousquet and Elisseeff (2002). Stability bounds depend on how the space of functions is searched by the algorithm (rather than the size of the function space), so it often yields more insightful bounds. These bounds are still not often directly useful due to large multiplicative constants (in our case a factor of 6), but they capture more closely the scalability relationship of a particular algorithm with respect to important quantities in the learning process. The calculation required for an algorithmic stability bound is to show that the empirical error will not dramatically change by altering or removing one of the training examples and re-running the algorithm. There are many different ways to measure the stability of an algorithm; most of the bounds presented here use a specific type of algorithmic stability (pointwise hypothesis stability) so that the bounds scale correctly with the number of training examples m.

Section 4.1 presents a basic stability bound for sequential event prediction. Section 4.2 presents a uniform VC bound for classification with max-score classifiers. Section 4.3 provides notation. Section 4.4 presents another basic stability bound for sequential event prediction, for a rule-based loss function. We then focus on stability bounds for the rule-based algorithms provided in Section 2. Specifically, Section 4.5 provides stability bounds for the large sample asymptotic regime (for both sequential event prediction and classification). Then we consider the new small m regime in Section 4.6, starting with stability bounds that formally show that minimum support thresholds can lead to better generalization (for both sequential event prediction and classification). From there, we present small sample bounds for the adjusted confidence algorithm, for classification and (separately) for sequential event prediction.

We note that the space of possible baskets (up to a maximum size) is a combinatorially large, discrete space. Because the space is discrete, all probability estimates converge to the true probabilities, which means that an algorithm that is statistically consistent can be obtained by estimating $p(b|B)$ directly for the current basket B. If m is large, prediction is easy. The difficult part is when we have only enough data to well estimate conditionals that are much smaller, $P(b|a), a \subset B$. That is the problem we are concerned with. Consistency does not imply anything about generalization bounds for the finite sample case.

4.1 General Stability Bound for Sequential Event Prediction

In this section we provide a basic stability-based bound for sequential event prediction, by analogy with Theorem 17 of Bousquet and Elisseeff (2002) (B&E).

We define a sequential event prediction algorithm producing f_S to have *strong sequential event prediction stability* β (by analogy with B&E Definition 15) if the following holds:

$$\forall S \in \mathcal{D}^m, \forall i \in \{1,...,m\}$$
$$\| \max_{t=0,...,T_z-1} |f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\}, z_{\cdot,t+1}) - f_{S^{/i}}(\{z_{\cdot,1},\ldots,z_{\cdot,t}\}, z_{\cdot,t+1})| \|_\infty \leq \beta,$$

where the ∞-norm is over baskets. A definition we will use from B&E is as follows: an algorithm producing function f_S with *uniform stability* β' obeys:

$$\forall S, \forall i \in \{1,...,m\}, \|\ell(f_S,\cdot) - \ell(f_{S^{/i}},\cdot)\|_\infty \leq \beta'.$$

Let us define a modified loss function. Let symbol Δ temporarily denote $f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},z_{\cdot,t+1}) - \max_{b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t}\}} f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b)$ in the expression below. The loss is:

$$\ell_\gamma(f_S,z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1}\begin{cases}1 & \text{if } \Delta \leq 0\\ 1-\frac{1}{\gamma}\Delta & \text{if } 0\leq\Delta\leq\gamma\\ 0 & \text{if } \Delta\geq\gamma.\end{cases}$$

The empirical error and leave-one-out error defined for this loss are:

$$\begin{aligned}\mathrm{EmpErr}_\gamma(f_S,z_i) &:= \frac{1}{m}\sum_{i=1}^m \ell_\gamma(f_S,z_i),\\ \mathrm{LooErr}_\gamma(f_S,z_i) &:= \frac{1}{m}\sum_{i=1}^m \ell_\gamma(f_{S^{/i}},z_i).\end{aligned}$$

Lemma 1 *A sequential event prediction algorithm producing f_S with strong sequential event prediction stability β has uniform stability $2\beta/\gamma$ with respect to the loss function ℓ_γ.*

Proof

$$\begin{aligned}&|\ell_\gamma(f_S,z) - \ell_\gamma(f_{S^{/i}},z)|\\ &\leq \frac{1}{T_z}\sum_{t=0}^{T_z-1}\frac{1}{\gamma}\left|\left[f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},z_{\cdot,t+1}) - \max_{b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t}\}} f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b)\right]\right.\\ &\quad\left. - \left[f_{S^{/i}}(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},z_{\cdot,t+1}) - \max_{b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t}\}} f_{S^{/i}}(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b)\right]\right|\\ &\leq \frac{1}{\gamma}\frac{1}{T_z}\sum_{t=0}^{T_z-1}\left[|f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},z_{\cdot,t+1}) - f_{S^{/i}}(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},z_{\cdot,t+1})| + \right.\\ &\quad\left.\left|\max_{b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t}\}} f_S(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b) - \max_{b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t}\}} f_{S^{/i}}(\{z_{\cdot,1},\ldots,z_{\cdot,t}\},b)\right|\right]\\ &\quad\leq \frac{1}{\gamma}2\beta.\end{aligned}$$

The first inequality uses the Lipschitz property of the loss, as well as an upper bound from moving the absolute values inside the sum. The third inequality uses the strong stability with respect to f_S. ■

The following theorem is analogous to Theorem 17 in B&E, for sequential event prediction. The proof is a direct application of Theorem 12 of B&E to the sequential event prediction loss, combined with Lemma 1.

Theorem 2 *Let f_S be a sequential event prediction algorithm with sequential event stability β. Then for all $\gamma > 0$ and any $m \geq 1$ and any $\delta \in (0,1)$ with probability at least $1-\delta$ over the random draw of sample S,*

$$\text{TrueErr}(f_S) \leq \text{EmpErr}_\gamma(f_S) + \frac{4\beta}{\gamma} + \left(8m\frac{\beta}{\gamma}+1\right)\sqrt{\frac{\ln(1/\delta)}{2m}}$$

and with probability at least $1-\delta$ over the random draw of sample S,

$$\text{TrueErr}(f_S) \leq \text{LooErr}_\gamma(f_S) + \frac{4\beta}{\gamma} + \left(8m\frac{\beta}{\gamma}+1\right)\sqrt{\frac{\ln(1/\delta)}{2m}}.$$

As with classification algorithms, the type of stability one would need to apply these bounds can be quite difficult to achieve, as it requires that the change in the model is small for any training set when any example is removed. This is particularly difficult to achieve when the sample size is somewhat small. For the association rule bounds, we know that uniform stability is not possible for many algorithms that perform well. However, there are some algorithms that do exhibit stronger stability, as we will discuss.

4.2 Classification with Association Rules: A Uniform Bound

In the classification problem, each basket receives a single label that is one of two possible labels $\{+1,-1\}$. This contrasts with sequential event prediction where there is a sequence of labels, one for each item in the basket as it arrives. For classification, we represent basket x as a binary vector, where entry j is 1 if item j is in the basket. We sample baskets with labels, $z=(x,y)$, where $x \in 2^{\mathcal{X}}$ is a set of items (or, equivalently, a binary feature vector) and $y \in \{-1,1\}$ is the corresponding label. Each labeled basket z is chosen randomly (iid) from a fixed (but unknown) probability distribution $\mathcal{D}$ over baskets and labels. Given a training set S of m labeled baskets, we wish to construct a classifier that can assign the correct label to new, unlabeled baskets. We begin by defining a scoring function $g : A \times \{-1,1\} \to \mathbb{R}$ that assigns a score $g(a,y)$ to a rule $a \to y$. The set of left-hand sides A can be any collection of itemsets so long as every $x \in 2^{\mathcal{X}}$ contains at least one $a \in A$. We define a *valid* scoring function as one where $\forall a \in A,\ g(a,1) \neq g(a,-1)$ and $\forall a_1, a_2 \in A,\ \max_{y\in\{-1,1\}} g(a_1,y) \neq \max_{y\in\{-1,1\}} g(a_2,y)$, that is, there are no ties. The validity requirement will be discussed in the following paragraph. Define G to be the class of all valid scoring functions. We now define a class of decision functions that use a valid scoring function $g \in G$ to provide a label to a basket x, $f_g : 2^{\mathcal{X}} \to \{-1,1\}$. The decision function assigns the label corresponding to the highest scoring rule whose left-hand side is contained in x. Specifically,

$$f_g(x) = \operatorname*{argmax}_{y\in\{-1,1\}} \max_{a\in A, a\subseteq x} g(a,y). \tag{3}$$

We call such a classifier a "max-score association rule classifier" (or "decision list") because it uses the association rule with the maximum score to perform the classification. Let $\mathcal{F}_{\text{maxscore}}$ be the class of all max-score association rule classifiers: $\mathcal{F}_{\text{maxscore}} := \{f_g : g \in G\}$. We will bound the VC dimension of class $\mathcal{F}_{\text{maxscore}}$. By definition, the VC dimension is the size of the largest set of baskets to which arbitrary labels can be assigned using some $f_g \in \mathcal{F}_{\text{maxscore}}$; it is the size of the largest set that can be shattered.

The argmax in (3) is unique because g is valid, thus there are no ties. If ties are allowed but broken randomly, arbitrary labels can be realized with some probability, for example by taking

$g(a,y) = 0$ for all a and y. In this case the VC dimension can be considered to be infinite, which motivates our definition of a valid scoring function. This problem actually happens with any classification problem where function $f(x) = 0\ \forall x$ is within the hypothesis space, thereby allowing all points to sit on the decision boundary. Our definition of validity is equivalent to one in which ties are allowed but are broken deterministically using a pre-determined ordering on the rules. In practice, ties are generally broken in a deterministic way by the computer, so the inclusion of the function $f = 0$ is not problematic.

The true error of the max-score association rule classifier is the expected misclassification error:

$$\text{TrueErrClass}(f_g) := \mathbb{E}_{(x,y)\sim\mathcal{D}} \mathbb{1}_{[f_g(x)\neq y]}. \tag{4}$$

The empirical error is the average misclassification error over a training set of m baskets:

$$\text{EmpErrClass}(f_g) := \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{[f_g(x_i)\neq y_i]}.$$

The main result of this subsection is the following theorem, which indicates that the size of the allowed set of left-hand sides may influence generalization.

Theorem 3 *(VC Dimension for Classification)*
The VC dimension h of the set of max-score classifiers is equal to the size of the allowed set of left hand sides of rules:

$$\text{VCdim}(\mathcal{F}_{\text{maxscore}}) := h := |A|.$$

From this theorem, classical results such as those of Vapnik (1999, Equations 20 and 21) can be directly applied to obtain a generalization bound:

Corollary 4 *(Uniform Generalization Bound for Classification)*
With probability at least $1-\delta$ *the following holds simultaneously for all* $f_g \in \mathcal{F}_{\text{maxscore}}$*:*

$$\text{TrueErrClass}(f_g) \leq \text{EmpErrClass}(f_g) + \frac{\varepsilon}{2}\left(1+\sqrt{1+\frac{4\text{EmpErrClass}(f_g)}{\varepsilon}}\right),$$

$$\textit{where } \varepsilon = 4\frac{|A|\left(\ln\frac{2m}{|A|}+1\right)-\ln\delta}{m}.$$

Note 1 (on uniform bounds): The result of Theorem 3 holds generally, well beyond the simple adjusted confidence or max confidence, min support algorithms. Those two algorithms correspond to specific choices of the scoring function g: the adjusted confidence algorithm takes $g(a,y) = f_{S,K}(a,y)$, and the max confidence, min support algorithm takes $g(a,y) = \bar{f}_{S,\theta}(a,y)$. We could use other strategies to choose g, for example, choosing $f_g \in \mathcal{F}$ to minimize an empirical risk (similar to what we do in Letham et al., 2013c).

Note 2 (on replacing itemsets with general boolean operators): Although in this paper we restrict our attention to left-hand sides that are sets of items (*e.g.*, "apples and oranges"), association rules can be constructed using the boolean operators AND, OR, and NOT (*e.g.*, "apples or oranges but not bananas"). In this case, the left-hand sides of rules are not contained in x, rather they are *true with respect to* x. By replacing "contained in x" with "true with respect to x" in the first half of the

proof of Theorem 3 (in Section 5), it can be seen that $h \leq |A|$ even when A contains general boolean association rules. Thus the bound in Corollary 4 extends to boolean operators.

Note 3 (dependence on $|A|$): We can use a standard argument involving Hoeffding's inequality and the union bound over elements of $\mathcal{F}_{\text{maxscore}}$ to obtain that with probability at least $1-\delta$, the following holds for all $f_g \in \mathcal{F}_{\text{maxscore}}$:

$$\text{TrueErrClass}(f_g) \leq \text{EmpErrClass}(f_g) + \sqrt{\frac{1}{2m}\left(\ln(2|\mathcal{F}_{\text{maxscore}}|) + \ln\frac{1}{\delta}\right)}.$$

The value of $|\mathcal{F}_{\text{maxscore}}|$ is at most $2^{|A|}$. This is because there are $|A|$ ways to determine $\max_{a\in A, a\subseteq x} g(a,y)$, and there are 2 ways to determine the argmax over y. The bound then depends on $\sqrt{|A|}$ (as classical VC bounds would also give, using Theorem 3), but not $\log|A|$. Note that the bound is meaningful when $|A| < m$ so that $2^{|A|} < 2^m$.

Note 4 (on reducing $|A|$): It is possible that many of the possible left-hand sides in $|A|$ are realized with zero probability. (This depends on the unknown probability distribution that the examples are drawn from.) Because of this, if we are willing to redefine A to include only realizable left-hand sides, $|A|$ can be replaced in the bound by $|\mathcal{A}|$, where $\mathcal{A} = \{a \in A : P_z(a \subseteq x) > 0\}$ are the itemsets that have some probability of being chosen.

4.3 Notation for Algorithmic Stability Bounds

We will now introduce the notation that will be used for the algorithmic stability bounds, first for classification and then for sequential event prediction.

4.3.1 Notation for Classification Bounds

Recall that we sample $z = (x,y)$ where $x \in 2^{\mathcal{X}}$ is a set of items and $y \in \{-1,1\}$ is the corresponding label. Each z is sampled randomly (iid) according to a distribution $\mathcal{D}$ over the space $2^{\mathcal{X}} \times \{-1,1\}$. The adjusted confidence algorithm uses the training set S of m iid baskets to compute the adjusted confidences $f_{S,K}$ and find a rule that will be used to label the basket. We use $z = (x,y)$ to refer to a general labeled basket, and $z_i = (x_i, y_i)$ to refer specifically to the i^{th} labeled basket in the training set. We define a *highest-scoring-correct* rule for x as a rule with the highest adjusted confidence that predicts the correct label y. The left-hand side of a highest-scoring-correct rule obeys:

$$a^+_{SxK} \in \operatorname*{argmax}_{a\subseteq x, a\in A} f_{S,K}(a,y) = \operatorname*{argmax}_{a\subseteq x, a\in A} \frac{\#(a\cup y)}{\#a+K},$$

where $K \geq 0$. If more than one rule is tied for the maximum adjusted confidence, one can now be chosen randomly. If the true label y is not found in the training set, then the confidence of all rules with y on the right-hand side will be 0, and we take $\varnothing \rightarrow y$ as the maximizing rule. We define a *highest-scoring incorrect* rule for x as a rule with the highest adjusted confidence that predicts the incorrect label $-y$, so the left-hand side obeys:

$$a^-_{SxK} \in \operatorname*{argmax}_{a\subseteq x, a\in A} f_{S,K}(a,-y) = \operatorname*{argmax}_{a\subseteq x, a\in A} \frac{\#(a\cup -y)}{\#a+K}.$$

Again, if the label $-y$ is not found in the training set, we take $\varnothing \to -y$ as the maximizing rule. Otherwise, ties are broken randomly.

A misclassification error is made for labeled basket z when the highest-scoring-correct rule, $a^+_{SxK} \to y$, has a lower adjusted confidence than the highest-scoring incorrect rule $a^-_{SxK} \to -y$. As discussed earlier, we will measure this difference in adjusted confidence values with respect to a reference K_r in order to allow comparisons with different values of K. We will take $K_r \geq 0$. This leads to the definition of the 0-1 loss for classification:

$$\ell^{\text{class}}_{0-1,K_r}(f_{S,K},z) := \begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{SxK},y) - f_{S,K_r}(a^-_{SxK},-y) \leq 0 \\ 0 & \text{otherwise.} \end{cases}$$

The term $f_{S,K_r}(a^+_{SxK},y) - f_{S,K_r}(a^-_{SxK},-y)$ is the "margin" of example z (that is, the gap in score between the predictions for the two classes, see also Shen and Wang, 2007).

We will now define the true error which, when $K = K_r$, is a specific case of TrueErrClass defined in (4). (The function g is chosen using the data set, and it is $f_{S,K}$.) The true error is an expectation of a loss function with respect to $\mathcal{D}$, and is a random variable since the training set S is random, $S \sim \mathcal{D}^m$.

$$\text{TrueErrClass}(f_{S,K},K_r) := \mathbb{E}_{z\sim\mathcal{D}}\ell^{\text{class}}_{0-1,K_r}(f_{S,K},z).$$

We approximate the true error using a different loss $\ell^{\text{class}}_{\gamma,K_r}$ that is a continuous upper bound on the 0-1 loss $\ell^{\text{class}}_{0-1,K_r}$. It is defined with respect to K_r and another real-valued parameter $\gamma > 0$ as follows:

$$\ell^{\text{class}}_{\gamma,K_r}(f_{S,K},z) := c_\gamma(f_{S,K_r}(a^+_{SxK},y) - f_{S,K_r}(a^-_{SxK},-y)),$$

where $c_\gamma : \mathbb{R} \to [0,1]$,

$$c_\gamma(y) = \begin{cases} 1 & \text{for } y \leq 0 \\ 1 - y/\gamma & \text{for } 0 \leq y \leq \gamma \\ 0 & \text{for } y \geq \gamma. \end{cases}$$

As γ approaches 0, loss c_γ approaches the standard 0-1 loss. Also, $\ell^{\text{class}}_{0-1,K_r}(f_{S,K},z) \leq \ell^{\text{class}}_{\gamma,K_r}(f_{S,K},z)$. We define $\text{TrueErrClass}_\gamma$ using this loss:

$$\text{TrueErrClass}_\gamma(f_{S,K},K_r) = \mathbb{E}_{z\sim\mathcal{D}}\ell^{\text{class}}_{\gamma,K_r}(f_{S,K},z),$$

where $\text{TrueErrClass} \leq \text{TrueErrClass}_\gamma$. The generalization bounds for classification will bound TrueErrClass by considering the difference between $\text{TrueErrClass}_\gamma$ and its empirical counterpart that we will soon define. For training basket x_i, the left-hand side of a highest-scoring-correct rule obeys:

$$a^+_{Sx_iK} \in \operatorname*{argmax}_{a\subseteq x_i, a\in A} f_{S,K}(a,y_i),$$

and the left-hand side of a highest-scoring-incorrect rule obeys:

$$a^-_{Sx_iK} \in \operatorname*{argmax}_{a\subseteq x_i, a\in A} f_{S,K}(a,-y_i).$$

The empirical error is an average of the loss over the baskets:

$$\text{EmpErrClass}_\gamma(f_{S,K},K_r) := \frac{1}{m}\sum_{i=1}^{m}\ell^{\text{class}}_{\gamma,K_r}(f_{S,K},z_i).$$

For the max confidence, min support algorithm, we substitute θ where K appears in the notation. For instance, for general labeled basket $z = (x, y)$, we analogously define:

$$\begin{aligned}
a^+_{Sx\theta} &\in \operatorname*{argmax}_{a\subseteq x, a\in A} \bar{f}_{S,\theta}(a, y), \\
a^-_{Sx\theta} &\in \operatorname*{argmax}_{a\subseteq x, a\in A} \bar{f}_{S,\theta}(a, -y), \\
\ell^{\text{class}}_{0-1,K_r}(\bar{f}_{S,\theta}, z) &= \begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{Sx\theta}, y) - f_{S,K_r}(a^-_{Sx\theta}, -y) \le 0 \\ 0 & \text{otherwise,} \end{cases} \\
\ell^{\text{class}}_{\gamma,K_r}(\bar{f}_{S,\theta}, z) &= c_\gamma(f_{S,K_r}(a^+_{Sx\theta}, y) - f_{S,K_r}(a^-_{Sx\theta}, -y)),
\end{aligned}$$

and $\text{TrueErrClass}(\bar{f}_{S,\theta}, K_r)$ and $\text{TrueErrClass}_\gamma(\bar{f}_{S,\theta}, K_r)$ are defined analogously as expectations of the losses, and $\text{EmpErrClass}_\gamma(\bar{f}_{S,\theta}, K_r)$ is again an average of the loss over the training baskets.

4.3.2 NOTATION FOR SEQUENTIAL EVENT PREDICTION BOUNDS

The notation and the bounds for sequential event prediction are similar to those of classification, the main differences being an additional index t to denote the different time steps, and a set of possible incorrect recommendations in the place of the single incorrect label $-y$. As defined in Section 3, a basket z consists of an ordered (permuted) set of items, $z \in 2^{\mathcal{X}} \times \Pi$, where $2^{\mathcal{X}}$ is the set of all subsets of $\mathcal{X}$, and Π is the set of permutations over at most $|\mathcal{X}|$ elements.[2] We have a training set of m baskets $S = \{z_i\}_{1\ldots m}$ that are the customer's past orders. Denote $z \sim \mathcal{D}$ to mean that basket z is drawn randomly (iid) according to distribution $\mathcal{D}$ over the space of possible items in baskets and permutations over those items, $2^{\mathcal{X}} \times \Pi$. The t^{th} item added to the basket is written $z_{\cdot,t}$, where the dot is just a placeholder for the generic basket z. The t^{th} element of the i^{th} basket in the training set is written $z_{i,t}$. We define the number of items in basket z by T_z, that is, $T_z := |z|$.

For sequential event prediction, a highest-scoring-correct rule is a highest scoring rule that has the next item $z_{\cdot,t+1}$ on the right. The left-hand side a^+_{SztK} of a highest-scoring-correct rule obeys:

$$a^+_{SztK} \in \operatorname*{argmax}_{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\}, a\in A} f_{S,K}(a, z_{\cdot,t+1}).$$

If $z_{\cdot,t+1}$ has never been purchased, the adjusted confidence for all rules $a \to z_{\cdot,t+1}$ is 0, and we choose the maximizing rule to be $\varnothing \to z_{\cdot,t+1}$. Also at time 0 when the basket is empty, the maximizing rule is $\varnothing \to z_{\cdot,t+1}$.

The algorithm incurs an error when it recommends an incorrect item. A highest-scoring-incorrect rule is a highest scoring rule that does not have $z_{\cdot,t+1}$ on the right. It is denoted $a^-_{SztK} \to b^-_{SztK}$, and obeys:

$$[a^-_{SztK}, b^-_{SztK}] \in \operatorname*{argmax}_{\substack{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\}, a\in A \\ b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t+1}\}}} f_{S,K}(a, b).$$

If there is more than one highest-scoring rule, one is chosen at random (with the exception that all incorrect rules are tied at zero adjusted confidence, in which case the left side is taken as $\varnothing$ and the right side is chosen randomly). At time $t = 0$, the left side is again $\varnothing$. The adjusted confidence algorithm determines a^+_{SztK}, a^-_{SztK}, and b^-_{SztK}, whereas nature chooses $z_{\cdot,t+1}$.

2. Even though we define an order for the basket for this discussion of prediction, we are still using the undirected adjusted confidence to make recommendations rather than the directed version introduced in Section 2. The results can be trivially extended to the directed case.

If the adjusted confidence of the rule $a^-_{S_{zt}K} \to b^-_{S_{zt}K}$ is larger than that of $a^+_{S_{zt}K} \to z_{\cdot,t+1}$, it means that the algorithm recommended the wrong item. The loss function below, which is the same as the one in Section 3 but with the algorithm built into it, again counts the proportion of times this happens for each basket, and is defined with respect to K_r.

$$\ell_{0-1,K_r}(f_{S,K},z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1}\begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{S_{zt}K},z_{\cdot,t+1}) - f_{S,K_r}(a^-_{S_{zt}K},b^-_{S_{zt}K}) \le 0 \\ 0 & \text{otherwise.}\end{cases}$$

The true error for sequential event prediction is an expectation of the loss:

$$\text{TrueErr}(f_{S,K},K_r) := \mathbb{E}_{z\sim\mathcal{D}}\ell_{0-1,K_r}(f_{S,K},z).$$

We create an upper bound for the true error by using a different loss ℓ_{γ,K_r} that is a continuous upper bound on the 0-1 loss ℓ_{0-1,K_r}. It is defined analogously to classification, with respect to K_r and c_γ:

$$\ell_{\gamma,K_r}(f_{S,K},z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1} c_\gamma(f_{S,K_r}(a^+_{S_{zt}K},z_{\cdot,t+1}) - f_{S,K_r}(a^-_{S_{zt}K},b^-_{S_{zt}K})).$$

It is true that $\ell_{0-1,K_r}(f_{S,K},z) \le \ell_{\gamma,K_r}(f_{S,K},z)$. We define TrueErr_γ:

$$\text{TrueErr}_\gamma(f_{S,K},K_r) := \mathbb{E}_{z\sim\mathcal{D}}\ell_{\gamma,K_r}(f_{S,K},z),$$

where $\text{TrueErr} \le \text{TrueErr}_\gamma$. The first set of results for sequential event prediction below bound TrueErr by considering the difference between TrueErr_γ and its empirical counterpart that we will soon define.

For the specific training basket z_i, the left-hand side $a^+_{S_{z_i}tK}$ of a highest-scoring-correct rule at time t obeys :

$$a^+_{S_{z_i}tK} \in \operatorname*{argmax}_{a\subseteq\{z_{i,1},\ldots,z_{i,t}\},a\in A} f_{S,K}(a,z_{i,t+1}),$$

similarly, a highest-scoring-incorrect rule for z_i at time t has:

$$[a^-_{S_{z_i}tK},b^-_{S_{z_i}tK}] \in \operatorname*{argmax}_{\substack{a\subseteq\{z_{i,1},\ldots,z_{i,t}\},a\in A\\ b\in\mathcal{X}\setminus\{z_{i,1},\ldots,z_{i,t+1}\}}} f_{S,K}(a,b).$$

The empirical error is defined as:

$$\text{EmpErr}_\gamma(f_{S,K},K_r) := \frac{1}{m}\sum_{\text{baskets } i=1}^{m}\ell_{\gamma,K_r}(f_{S,K},z_i).$$

For the max confidence, min support algorithm, we again substitute θ where K appears in the notation. For example, we define:

$$\begin{aligned}
a^+_{S_{zt}\theta} &\in \operatorname*{argmax}_{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\},a\in A} \bar{f}_{S,\theta}(a,z_{\cdot,t+1}),\\
\left[a^-_{S_{zt}\theta},b^-_{S_{zt}\theta}\right] &\in \operatorname*{argmax}_{\substack{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\},a\in A\\ b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t+1}\}}} \bar{f}_{S,\theta}(a,b),\\
\ell_{0-1,K_r}(\bar{f}_{S,\theta},z) &:= \frac{1}{T_z}\sum_{t=0}^{T_z-1}\begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{S_{zt}\theta},z_{\cdot,t+1}) - f_{S,K_r}(a^-_{S_{zt}\theta},b^-_{S_{zt}\theta}) \le 0\\ 0 & \text{otherwise,}\end{cases}\\
\ell_{\gamma,K_r}(\bar{f}_{S,\theta},z) &:= \frac{1}{T_z}\sum_{t=0}^{T_z-1} c_\gamma(f_{S,K_r}(a^+_{S_{zt}\theta},z_{\cdot,t+1}) - f_{S,K_r}(a^-_{S_{zt}\theta},b^-_{S_{zt}\theta})).
\end{aligned}$$

$\text{TrueErr}(\bar{f}_{S,\theta}, K_r)$ and $\text{TrueErr}_\gamma(\bar{f}_{S,\theta}, K_r)$ are expectations of the losses, and $\text{EmpErr}_\gamma(\bar{f}_{S,\theta}, K_r)$ is an average of the loss over the training baskets.

4.4 General Stability Bound for Sequential Event Prediction with Rule-Based Loss

This section contains a stability bound for sequential event prediction, by analogy with Theorem 17 of Bousquet and Elisseeff (2002), using the loss we just defined, which involves rules. We need to define what is meant by a rule-based sequential event prediction algorithm. To keep this definition general, we define an algorithm Alg to take as input a data set S, basket z, and item b^* (where b^* is the desired output for basket z), and have the algorithm output: (i) the left hand side of the algorithm's chosen rule to predict b^*, which we call $a^+_{S,z,b^*,Alg}$, (ii) the algorithm's chosen rule that predicts an item other than b^*, which is called $a^-_{S,z,b^*,Alg} \rightarrow b^-_{S,z,b^*,Alg}$.

We define $Alg : S, z, b^* \mapsto a^+_{S,z,b^*,Alg}, a^-_{S,z,b^*,Alg}, b^-_{S,z,b^*,Alg}$ to have *uniform rule stability* β for sequential event prediction with respect to K_r if:

$$\forall S, \forall z, \forall b^*, \text{ we have } |f_{S,K_r}(a^+_{S,z,b^*,Alg}, b^*) - f_{S,K_r}(a^+_{S^{/i},z,b^*,Alg}, b^*)| \leq \beta \text{ and}$$

$$|f_{S,K_r}(a^-_{S,z,b^*,Alg}, b^-_{S,z,b^*,Alg}) - f_{S,K_r}(a^-_{S^{/i},z,b^*,Alg}, b^-_{S^{/i},z,b^*,Alg})| \leq \beta.$$

That is, the algorithm is stable whenever (i) the adjusted confidence of the rules used to predict both b^* is not affected much by the removal of one training example, and (ii) when the adjusted confidence of the rule to predict something other than b^* is not affected much by the removal of one training example. We can then show:

Lemma 5 *A rule-based sequential event prediction algorithm with uniform rule stability* β *has uniform stability* $2\beta/\gamma$ *with respect to the loss function* ℓ_{γ,K_r}.

Proof

$$\begin{aligned}
&\left|\ell_{\gamma,K_r}\left(Alg(S,\cdot,\cdot),z\right) - \ell_{\gamma,K_r}\left(Alg(S^{/i},\cdot,\cdot),z\right)\right| \\
&= \left|\frac{1}{T_z}\sum_{t=0}^{T_z-1} c_\gamma\Big(f_{S,K_r}(a^+_{S,z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, z_{\cdot,t+1}) \right. \\
&\quad - f_{S,K_r}(a^-_{S,z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, b^-_{S,z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg})\Big) \\
&\quad - c_\gamma\Big(f_{S,K_r}(a^+_{S^{/i},z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, z_{\cdot,t+1}) \\
&\quad \left. - f_{S,K_r}(a^-_{S^{/i},z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, b^-_{S^{/i},z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg})\Big)\right| \\
&\leq \frac{1}{T_z,\gamma}\sum_{t=0}^{T_z-1}\Big|f_{S,K_r}(a^+_{S,z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, z_{\cdot,t+1}) \\
&\quad - f_{S,K_r}(a^-_{S^{/i},z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, z_{\cdot,t+1})\Big| \\
&\quad + \Big|f_{S,K_r}(a^-_{S,z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, b^-_{S,z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}) \\
&\quad - f_{S,K_r}(a^-_{S^{/i},z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg}, b^-_{S^{/i},z_{\cdot,1}\ldots z_{\cdot,t},z_{\cdot,t+1},Alg})\Big| \\
&\leq \frac{1}{\gamma}2\beta.
\end{aligned}$$

In the first inequality, we used the Lipschitz property of the loss, and properties of absolute values. In the second inequality, we used the definition of uniform rule stability for both absolute value terms with b^* being $z_{\cdot,t+1}$, and basket z being $z_{\cdot,1}...z_{\cdot,t}$. ■

Adapting the definitions in the previous subsection to Alg (rather than f_S), the following theorem is analogous to Theorem 17 in B&E, for the rule-based loss ℓ_{γ,K_r} for sequential event prediction. The proof is an application of Theorem 12 of B&E to the rule-based sequential event prediction loss, combined with Lemma 5.

Theorem 6 *Let Alg be a sequential event prediction algorithm with uniform rule stability β for sequential event stability. Then for all $\gamma > 0$ and any $m \geq 1$ and any $\delta \in (0,1)$ with probability at least $1-\delta$ over the random draw of sample S,*

$$\text{TrueErr}(Alg, K_r) \leq \text{EmpErr}_\gamma(Alg, K_r) + \frac{4\beta}{\gamma} + \left(8m\frac{\beta}{\gamma} + 1\right)\sqrt{\frac{\ln(1/\delta)}{2m}}$$

and with probability at least $1-\delta$ over the random draw of sample S,

$$\text{TrueErr}(Alg, K_r) \leq \text{LooErr}_\gamma(Alg, K_r) + \frac{4\beta}{\gamma} + \left(8m\frac{\beta}{\gamma} + 1\right)\sqrt{\frac{\ln(1/\delta)}{2m}}.$$

We now focus our attention back to the rule-based algorithms from Section 2, and derive a variety of bounds for these algorithms.

4.5 Generalization Analysis for Large m

The choice of minimum support threshold θ or the choice of parameter K matters mainly in the regime where m is small. For the max confidence, min support algorithm, when m is large, then all (realizable) itemsets have appeared more times than the minimum support threshold with high probability. For the adjusted confidence algorithm, when m is large, prediction ability is guaranteed as follows.

Theorem 7 *(Generalization Bound for Adjusted Confidence Algorithm, Large m)*
For set of rules A, $K \geq 0$, $K_r \geq 0$, with probability at least $1-\delta$ (with respect to training set $S \sim \mathcal{D}^m$),

$$\text{TrueErr}(f_{S,K}, K_r) \leq \text{EmpErr}_\gamma(f_{S,K}, K_r) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + 6\beta\right]}$$
$$\text{where } \beta = \frac{2|\mathcal{A}|}{\gamma}\left[\frac{1}{(m-1)p_{\min A} + K} + \frac{|K_r - K|\frac{m}{m+K}}{(m-1)p_{\min A} + K_r}\right] + O\left(\frac{1}{m^2}\right),$$

and where $\mathcal{A} = \{a \in A : P_z(a \subseteq z) > 0\}$ are the itemsets that have some probability of being chosen. Out of these, any itemset that is the least likely to be chosen has probability $p_{\min A}$:

$$p_{\min A} := \min_{a \in \mathcal{A}} P_{z\sim\mathcal{D}}(a \subseteq z).$$

As a corollary, the same result holds for classification, replacing $\text{TrueErr}(f_{S,K}, K_r)$ with $\text{TrueErrClass}(f_{S,K}, K_r)$ and $\text{EmpErr}_\gamma(f_{S,K}, K_r)$ with $\text{EmpErrClass}_\gamma(f_{S,K}, K_r)$.

A special case is where $K_r = K = 0$: the algorithm chooses the rule with maximum confidence, and accuracy is then judged by the difference in confidence values between the highest-scoring-incorrect rule and the highest-scoring-correct rule. The bound reduces to:

Corollary 8 *(Generalization Bound for Maximum Confidence Setting, Large m)*
With probability at least $1-\delta$ *(with respect to* $S \sim \mathcal{D}^m$*),*

$$\text{TrueErr}(f_{S,0}, 0) \leq \text{EmpErr}_\gamma(f_{S,0}, 0) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + \frac{12|\mathcal{A}|}{\gamma(m-1)p_{\min A}}\right] + O\left(\frac{1}{m^2}\right)}.$$

Again the result holds for classification with appropriate substitutions. The use of the pointwise hypothesis stability within this proof is the key to providing a decay of order $\sqrt{(1/m)}$. Now that this bound is established, we move to the small sample case, where the minimum support is the force that provides generalization.

4.6 Generalization Analysis for Small m

The first small sample result is a general bound for the max confidence, min support algorithm, which holds for both classification and sequential event prediction. The max confidence, min support algorithm has uniform stability, which is a stronger kind of stability than pointwise hypothesis stability. This result strengthens the one in the conference version of this work (Rudin et al., 2011), where we used the bound for pointwise hypothesis stability; uniform stability implies pointwise hypothesis stability, so the result in the conference version follows automatically.

Theorem 9 *(Generalization Bound for Max Confidence, Min Support)*
For $\theta \geq 1$, $K_r \geq 0$, *with probability at least* $1-\delta$ *(with respect to* $S \sim \mathcal{D}^m$*),* $m > \theta$*,*

$$\text{TrueErr}(\bar{f}_{S,\theta}, K_r) \leq \text{EmpErr}_\gamma(\bar{f}_{S,\theta}, K_r) + 2\beta + (4m\beta + 1)\sqrt{\frac{\ln 1/\delta}{2m}}$$
$$\text{where } \beta = \frac{2}{\gamma}\left[\frac{1}{\theta} + K_r\left(\frac{1}{\theta + K_r}\right)\left(1 + \frac{1}{\theta}\right)\right].$$

Note that $|A|$ does not appear in the bound. For classification, $\text{TrueErr}(\bar{f}_{S,\theta}, K_r)$ is replaced by $\text{TrueErrClass}(\bar{f}_{S,\theta}, K_r)$ and $\text{EmpErr}_\gamma(\bar{f}_{S,\theta}, K_r)$ is replaced by $\text{EmpErrClass}_\gamma(\bar{f}_{S,\theta}, K_r)$. Figure 2 shows β as a function of θ for several different values of K_r. The special case of interest is when $K_r = 0$, so that the loss is judged with respect to differences in confidence, as follows:

Corollary 10 *(Generalization Bound for Max Confidence, Min Support,* $K_r = 0$*)*
For $\theta \geq 1$, *with probability at least* $1-\delta$ *(with respect to* $S \sim \mathcal{D}^m$*),* $m > \theta$*,*

$$\text{TrueErr}(\bar{f}_{S,\theta}, 0) \leq \text{EmpErr}_\gamma(\bar{f}_{S,\theta}, 0) + \frac{4}{\gamma\theta} + \left(\frac{8m}{\gamma\theta} + 1\right)\sqrt{\frac{\ln 1/\delta}{2m}}.$$

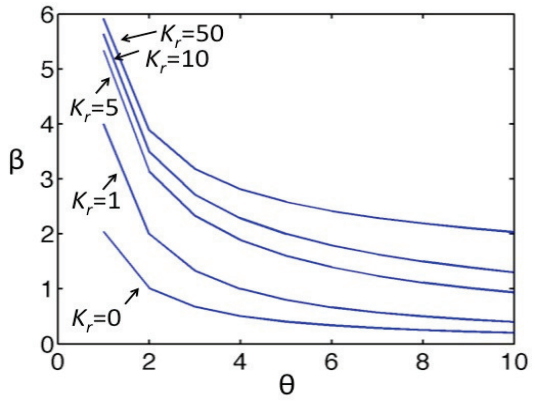


Figure 2: β vs. θ from Theorem 9, with $\gamma = 1$. The different curves are different values of $K_r = 0$, 1, 5, 10, 50 from bottom to top.

It is common to use a minimum support threshold that is a fraction of m, for instance, $\theta = 0.1 \times m$. In that case, the bound again scales with $\sqrt{(1/m)}$. Note that there is no generalization guarantee when $\theta = 0$; the minimum support threshold enables generalization in the small m case.

Now we discuss the adjusted confidence algorithm for small m setting. We present separate small sample bounds for classification and sequential event prediction.

Theorem 11 *(Generalization Bound for Adjusted Confidence Algorithm, Small m, For Classification Only) For $K > 0, K_r \geq 0$, with probability at least $1-\delta$,*

$$\text{TrueErrClass}(f_{S,K}, K_r) \leq \text{EmpErrClass}_\gamma(f_{S,K}, K_r) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + 6\beta\right]} \text{ where}$$

$$\begin{aligned}\beta &= \frac{2}{\gamma}\frac{1}{K}\left(1 - \frac{(m-1)p_{y,\min}}{m+K}\right) \\ &\quad + \frac{2}{\gamma}|K_r - K|\mathbb{E}_{\zeta \sim \text{Bin}(m-1, p_{y,\min})}\left[\frac{1}{K\left(\frac{\zeta}{m+K-\zeta}\right) + K_r}\left(\frac{m}{m+K} + \frac{1}{K}\left(1 - \frac{\zeta}{m+K}\right)\right)\right],\end{aligned}$$

where $p_{y,\min} = \min(P(y=1), P(y=-1))$ is the probability of the less popular label.

Again, $|A|$ does not appear in the bound, and generalization is provided by K, and the difference between K and K_r; the interpretation will be further discussed after we state the small sample bound for sequential event prediction.

In the proof of the following theorem, if we were to use the definitions established in Section 4.3.2, the bound does not simplify beyond a certain point and is difficult to read at an intuitive level. From that bound, it would not be easy to see what are the important quantities for the learning process, and how they scale. In what follows, we redefine the loss function slightly, so that it approximates a 0-1 loss from below instead of from above. This provides a concise and intuitive bound.

Define a *highest-scoring* rule $a^*_{SztK} \to b^*_{SztK}$ as a rule that achieves the maximum adjusted confidence, over all of the possible rules. It will either be equal to $a^+_{SztK} \to z_{\cdot,t+1}$ or $a^-_{SztK} \to b^-_{SztK}$, depending on which has the larger adjusted confidence:

$$[a^*_{SztK}, b^*_{SztK}] \in \operatorname*{argmax}_{\substack{a \subseteq \{z_{\cdot,1},\ldots,z_{\cdot,t}\}, a \in A \\ b \in \mathcal{X} \setminus \{z_{\cdot,1},\ldots,z_{\cdot,t}\}}} f_{S,K}(a,b).$$

Note that b^*_{SztK} can be equal to $z_{\cdot,t+1}$ whereas b^-_{SztK} cannot. The notation for $a^*_{Sz_itK}$ and $b^*_{Sz_itK}$ is similar, and the new loss is:

$$\ell^{\text{new}}_{0-1,K_r}(f_{S,K},z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1}\begin{cases} 1 & \text{if } f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^*_{SztK}, b^*_{SztK}) < 0 \\ 0 & \text{otherwise.}\end{cases}$$

By definition, the difference $f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^*_{SztK}, b^*_{SztK})$ can never be strictly positive. The continuous approximation is:

$$\ell^{\text{new}}_{\gamma,K_r}(f_{S,K},z) := \frac{1}{T_z}\sum_{t=0}^{T_z-1} c^{\text{new}}_\gamma(f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^*_{SztK}, b^*_{SztK})), \text{ where}$$

$$c^{\text{new}}_\gamma(y) = \begin{cases} 1 & \text{for } y \le -\gamma \\ -y/\gamma & \text{for } -\gamma \le y \le 0 \\ 0 & \text{for } y \ge 0. \end{cases}$$

As γ approaches 0, the c_γ loss approaches the 0-1 loss. We define $\text{TrueErr}^{\text{new}}_\gamma$ and $\text{EmpErr}^{\text{new}}_\gamma$ using this loss: $\text{TrueErr}^{\text{new}}_\gamma(f_{S,K},K_r) := \mathbb{E}_{z\sim\mathcal{D}}\ell^{\text{new}}_{\gamma,K_r}(f_{S,K},z)$, and $\text{EmpErr}^{\text{new}}_\gamma(f_{S,K},K_r) := \frac{1}{m}\sum_{i=1}^m \ell^{\text{new}}_{\gamma,K_r}(f_{S,K},z_i)$.

The minimum support threshold condition we used in Theorem 9 is replaced by a weaker condition on the support. This weaker condition has the benefit of allowing more rules to be used in order to achieve a better empirical error; however, it is more difficult to get a generalization guarantee. This support condition is derived from the fact that the adjusted confidence of the highest-scoring rule $a^*_{Sz_itK} \to b^*_{Sz_itK}$ exceeds that of the highest-scoring-correct rule $a^+_{Sz_itK} \to z_{i,t+1}$, which exceeds that of the marginal rule $\varnothing \to z_{i,t+1}$:

$$\frac{\#a^*_{Sz_itK}}{\#a^*_{Sz_itK}+K} \ge \frac{\#(a^*_{Sz_itK} \cup b^*_{Sz_itK})}{\#a^*_{Sz_itK}+K} \ge \frac{\#(a^+_{Sz_itK} \cup z_{i,t+1})}{\#a^+_{Sz_itK}+K} \ge \frac{\#z_{i,t+1}}{m+K}. \tag{5}$$

This leads to a lower bound on the support $\#a^*_{Sz_itK}$:

$$\#a^*_{Sz_itK} \ge K\left(\frac{\#z_{i,t+1}}{m+K-\#z_{i,t+1}}\right). \tag{6}$$

This is not a hard minimum support threshold, yet since the support generally increases as K increases, the bound will give a better guarantee for large K. Note that in the original notation, we would replace the condition (5) with $\frac{\#a^-_{Sz_itK}}{\#a^-_{Sz_itK}+K} \ge \frac{\#(a^-_{Sz_itK}\cup b^-_{Sz_itK})}{\#a^-_{Sz_itK}+K} \ge \frac{\#b^-_{Sz_itK}}{m+K}$ and proceed with analogous steps in the proof.

Theorem 12 *(Generalization Bound for Adjusted Confidence Algorithm, Small m) For $K > 0, K_r \ge 0$, with probability at least $1-\delta$,*

$$\text{TrueErr}^{\text{new}}_\gamma(f_{S,K},K_r) \le \text{EmpErr}^{\text{new}}_\gamma(f_{S,K},K_r) + \sqrt{\frac{1}{\delta}\left[\frac{1}{2m} + 6\beta\right]} \text{ where}$$

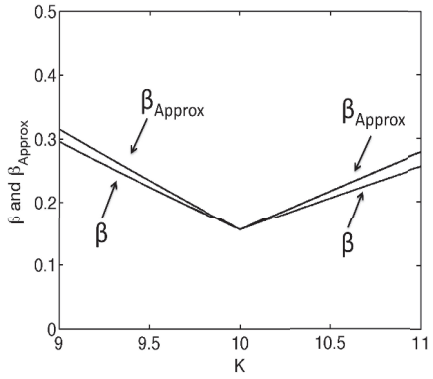


Figure 3: β and β_{Approx} vs. K, where $K_r = 10$, $p_{\min} = 0.3$, $m = 20$, $\gamma = 1$.

$$\begin{aligned}\beta &= \frac{2}{\gamma}\frac{1}{K}\left(1 - \frac{(m-1)p_{\min}}{m+K}\right) \\ &\quad + \frac{2}{\gamma}|K_r - K|\mathbb{E}_{\zeta\sim\text{Bin}(m-1,p_{\min})}\frac{1}{K\left(\frac{\zeta}{m+K-\zeta-1}\right)+K_r}\left(\frac{m}{m+K} + \frac{1}{K}\left(1 - \frac{\zeta}{m+K}\right)\right),\end{aligned}$$

and where $Q = \{x \in \mathcal{X} : P_{z\sim D}(x \in z) > 0\}$ *are the items that have some probability of being chosen by the customer. Out of these, any item that is the least likely to be chosen has probability* $p_{\min} := \min_{x\in Q} P_{z\sim\mathcal{D}}(x \in z)$.

The stability β has two main terms. The first term decreases generally as $1/K$. The second term arises from the error in measuring loss with K_r rather than K. In order to interpret β, consider the following approximation to the expectation in the bound, which assumes that m is large and that $m \gg K \gg 0$, and that $\zeta \approx mp_{\min}$:

$$\beta \approx \frac{2}{\gamma}\frac{1}{K}\left(1 - \frac{(m-1)p_{\min}}{m+K}\right) + \frac{2}{\gamma}|K_r - K|\frac{1}{K\frac{p_{\min}}{1-p_{\min}} + K_r}. \tag{7}$$

Intuitively, if either K is close to K_r or $p_{\min}$ is large (close to 1) then this term becomes small. Figure 3 shows an example plot of β and the approximation using (7), which we denote by β_{Approx}.

One can observe that if $K_r > K$, then both terms tend to improve (decrease) with increasing K. When $K_r < K$, then the two terms can compete as K increases.

4.7 Summary of Bounds

We have provided probabilistic guarantees on performance that show the following: 1) For large m, the association rule-based algorithms have a performance guarantee of the same order as other bounds for supervised learning. 2) For small m, the minimum support threshold guarantees generalization (at the expense of possibly removing important rules). 3) The adjusted confidence provides a weaker support threshold, allowing important rules to be used, while still being able to generalize. 4) All generalization guarantees depend on the way the goodness of the algorithm is measured (the choice of K_r in the loss function). 5) Important quantities in the learning process may include: $|A|$ or $|\mathcal{A}|$, K or θ, $p_{\min A}$ or $p_{\min}$ (or $p_{y,\min}$).

5. Proofs

In this section, we prove all results from Section 4.

Proof *(Of Theorem 3)* First we show that $h \leq |A|$. To do this, we must show that for any collection of baskets $x_1, \ldots, x_N$, $N > |A|$, there exists a corresponding set of labels $y_1, \ldots, y_N$ that cannot be realized by any max-score association rule classifier. For each x_i, we introduce a vector $\bar{x}_i$ of length $|A|$, where each element corresponds to an $a \in A$. The element of $\bar{x}_i$ corresponding to a is 1 if $a \subseteq x_i$ and 0 otherwise. Each vector $\bar{x}_i$ is an element of $\mathbb{R}^{|A|}$, so the collection of vectors $\bar{x}_1, \ldots, \bar{x}_N$ must be linearly dependent if $N > |A|$. By linear dependence and the fact that every $\bar{x}_i$ is non-zero and non-negative, there must exist coefficients c_i and disjoint, non-empty sets M_0 and M_1 such that:

$$\sum_{i \in M_0} c_i \bar{x}_i = \sum_{i \in M_1} c_i \bar{x}_i, \; c_i > 0. \tag{8}$$

Define $A_0 = \{a \in A : a \subseteq x_i \text{ for some } i \in M_0\}$ and $A_1 = \{a \in A : a \subseteq x_i \text{ for some } i \in M_1\}$. If $a \subseteq x_i$ for some $i \in M_0$, then the corresponding element of $\bar{x}_i$ will be 1 and the same element in the left part of (8) will be strictly positive. Then, (8) implies that $a \subseteq x_j$ for some $j \in M_1$. Thus, $A_0 \subseteq A_1$, and the reverse argument shows $A_1 \subseteq A_0$, so $A_0 = A_1$. There exists a left-hand side with maximum score, $a^* = \arg\max_{a \in A_0} \max_{y \in \{-1,1\}} g(a,y) = \arg\max_{a \in A_1} \max_{y \in \{-1,1\}} g(a,y)$. The label assigned to x_i, where i is in M_0 or M_1 and x_i contains itemset a^*, is $y^* = \arg\max_{y \in \{-1,1\}} g(a^*, y)$. Thus for at least one $i \in M_0$ and at least one $j \in M_1$, $f_g(x_i) = y^* = f_g(x_j)$. Set $y_i = -1$ for all $i \in M_0$ and $y_i = 1$ for all $i \in M_1$ and this set of labels cannot be realized, which shows that $h \leq |A|$.

We now show that this upper bound can be achieved by providing a set of $|A|$ baskets and finding elements of $\mathcal{F}_{\text{maxscore}}$ that can assign them arbitrary labels. Specifically, we list the elements of A as $a_1, \ldots, a_{|A|}$ and take $x_i = a_i$, for $i = 1, \ldots, |A|$. Thus each basket is one of the left-hand sides from the allowed set. The elements of A are not all the same size, and some elements of A may contain other elements; this could cause problems when we are constructing a max-score classifier that uniquely assigns a given label to each basket. To get around this, we will place the elements of A in order of increasing size. The possible sizes of elements of A are denoted $l_1, \ldots, l_L$, so that $l_1 < l_2 < \ldots < l_L$. We arrange the elements of A into sets based on their sizes: $S_k = \{i : |a_i| = l_k\}$, $k = 1, 2, \ldots, L$. We are now ready to construct a classifier f_g so that, given an arbitrary set of labels $\{y_i\}_i$, it can label the x_i's according to the y_i's. For all $i \in S_1$, we set $g(a_i, y_i) = c_1$, any positive number, and $g(a_i, -y_i) = 0$. Thus, for the corresponding x_i, $f_g(x_i) = y_i$. Similarly, for all $i \in S_2$, we set $g(a_i, y_i) = c_2$, $c_2 > c_1$, and $g(a_i, -y_i) = 0$. For any $i \in S_2$, it may be that there exists some $j \in S_1$ such that $a_j \subset x_i$. However, because $c_2 > c_1$, the rule with the maximum score will be "$a_i \to y_i$" and x_i is labeled as desired. In general, for any $i \in S_k$, we set $g(a_i, y_i) = c_k$, where $c_{k-1} < c_k < c_{k+1}$ and $g(a_i, -y_i) = 0$ to get $f_g(x_i) = y_i$. Because this set of $|A|$ examples can be arbitrarily labeled using elements of $\mathcal{F}_{\text{maxscore}}$, we have $h \geq |A|$, which combined with the previous result shows that $h = |A|$. ■

The remaining theorems are based on the algorithmic stability bounds of Bousquet and Elisseeff (2002) (B&E). Many of the proofs that we provide for classification are essentially identical to those for sequential event prediction. In these cases, the proofs are given for sequential event prediction, and afterwards the translation to classification is outlined. The proofs follow this outline: first, we show how differences in adjusted confidence values with respect to K_r can be translated into differences with respect to K (Lemma 15). Then we bound the difference in adjusted confidence values (Lemma 16) in terms of the support. Various lower bounds on the support are used to obtain

stability for each of the separate cases: large m (Theorem 7), small m for the max confidence, min support algorithm (Theorem 9, which uses uniform stability), small m for classification with the adjusted confidence algorithm (Theorem 11), and small m for sequential event prediction with the adjusted confidence algorithm (Theorem 12).

Following notation of Bousquet and Elisseeff (2002), the input space and output space are X and Y. Their training set is $S \in \bar{Z}^m$, $S = \{\bar{z}_1 = (x_1, y_1), \ldots, \bar{z}_m = (x_m, y_m)\}$. An algorithm is a function A from $\bar{Z}^m$ into $\mathcal{F} \subset Y^X$ which maps a learning set S onto a function A_S from X to Y. The loss is $\ell(f, \bar{z}) = c(f(x), y)$, where $c : Y \times Y \to \mathbb{R}_+$. $S^{/i}$ means to exclude the i^{th} example $\bar{z}_i$. B&E assume that $Y \subset \mathbb{R}$ but we believe this assumption is unnecessary. In any case, Y is empty for sequential event prediction. An algorithm A has *pointwise hypothesis stability* β with respect to the loss function ℓ if the following holds:

$$\forall\, i \in \{1, \ldots, m\},\ \mathbb{E}_{S \sim \mathcal{D}^m}[|\ell(A_S, \bar{z}_i) - \ell(A_{S/i}, \bar{z}_i)|] \leq \beta.$$

An algorithm A has *uniform stability* β with respect to the loss function ℓ if the following holds:

$$\forall\, S \in \bar{Z}^m, \forall\, i \in \{1, \ldots, m\},\ ||\ell(A_S, \cdot) - \ell(A_{S/i}, \cdot)||_\infty \leq \beta.$$

The empirical error is defined by:

$$R_{emp}(A, S) := \frac{1}{m} \sum_{i=1}^{m} \ell(A_S, \bar{z}_i)$$

and the true error is:

$$R(A, S) := \mathbb{E}_{\bar{z}}[\ell(A_S, \bar{z})].$$

We will use the following results that are based on ideas of Devroye and Wagner (1979).

Theorem 13 *(B&E Pointwise Hypothesis Stability Bound)(Bousquet and Elisseeff, 2002, Theorem 11, first part)*
For any learning algorithm A with pointwise hypothesis stability β *with respect to a loss function* ℓ*, such that the value of* ℓ *is at most M, we have with probability* $1 - \delta$*,*

$$R(A, S) \leq R_{emp}(A, S) + \sqrt{\frac{M^2 + 12Mm\beta}{2m\delta}}.$$

Theorem 14 *(B&E Uniform Stability Bound)(Bousquet and Elisseeff, 2002, Theorem 12, first part)*
For any learning algorithm A with uniform stability β *with respect to a loss function* ℓ*, such that the value of* ℓ *is at most M, we have with probability* $1 - \delta$ *over a random draw of S,*

$$R \leq R_{emp} + 2\beta + (4m\beta + M)\sqrt{\frac{\ln 1/\delta}{2m}}.$$

Translating B&E's notation to the adjusted confidence setting for sequential event prediction, $\bar{z}_i = x_i = z_i$, with $z_i \in 2^X \times \Pi$. For our problem, $f(x_i)$ is the value of the loss and the y_i's are not defined. In other words, $\ell(A_S, \bar{z}_i) = c(f(x_i), y_i) = f(x_i)$ which in our notation is equal to $\ell_{\gamma, K_r}(f_{S,K}, z_i)$. For the max confidence, min support setting, $\ell(A_S, \bar{z}_i)$ translates to $\ell_{\gamma, K_r}(\bar{f}_{S,\theta}, z_i)$. The adjusted confidence is bounded by 1 so $M = 1$.

The following lemma allows us to convert differences in adjusted confidence with respect to K_r into differences with respect to K.

Lemma 15 *(Conversion of Adjusted Confidence) For $K \geq 0$, $K_r \geq 0$, $0 \leq s_1 \leq S_1$, $0 \leq s_2 \leq S_2$*

$$\left|\frac{s_1}{S_1+K_r}-\frac{s_2}{S_2+K_r}\right| \leq \left|\frac{s_1}{S_1+K}-\frac{s_2}{S_2+K}\right|\left(1+\frac{|K_r-K|}{S_1+K_r}\right)+\left(\frac{|K_r-K|}{\tilde{S}+K_r}\right)\left(\frac{s_2}{S_2+K}\right)$$

where $\tilde{S}=\min(S_1,S_2)$.

Proof

$$\begin{aligned}
&\left|\frac{s_1}{S_1+K_r}-\frac{s_2}{S_2+K_r}\right| \\
&= \left|\frac{s_1}{S_1+K}-\frac{s_2}{S_2+K}+(-K_r+K)\left[\frac{s_1}{S_1+K}\left(\frac{1}{S_1+K_r}\right)-\frac{s_2}{S_2+K}\left(\frac{1}{S_2+K_r}\right)\right]\right| \\
&\leq \left|\frac{s_1}{S_1+K}-\frac{s_2}{S_2+K}\right|+|K_r-K|\left|\frac{s_1}{S_1+K}\left(\frac{1}{S_1+K_r}\right)-\frac{s_2}{S_2+K}\left(\frac{1}{S_2+K_r}\right)\right|. \qquad (9)
\end{aligned}$$

Taking just the second absolute value term:

$$\begin{aligned}
&\left|\frac{s_1}{S_1+K}\left(\frac{1}{S_1+K_r}\right)-\frac{s_2}{S_2+K}\left(\frac{1}{S_2+K_r}\right)\right| \\
&= \left|\frac{s_1}{S_1+K}\left(\frac{1}{S_1+K_r}\right)-\frac{s_2}{S_2+K}\left(\frac{1}{S_1+K_r}\right)+\frac{s_2}{S_2+K}\left(\frac{1}{S_1+K_r}\right)-\frac{s_2}{S_2+K}\left(\frac{1}{S_2+K_r}\right)\right| \\
&\leq \left|\frac{s_1}{S_1+K}-\frac{s_2}{S_2+K}\right|\left|\frac{1}{S_1+K_r}+\frac{s_2}{S_2+K}\right|\left|\frac{1}{S_1+K_r}-\frac{1}{S_2+K_r}\right| \\
&\leq \left|\frac{s_1}{S_1+K}-\frac{s_2}{S_2+K}\right|\left|\frac{1}{S_1+K_r}+\frac{s_2}{S_2+K}\right|\left|\frac{1}{\tilde{S}+K_r}\right|.
\end{aligned}$$

Putting this back into (9) yields the statement. ■

The next results bound the difference in the highest adjusted confidence values when the basket z_i is removed from S. We require some additional notation in order to exclude basket i. Denote $\#^{/i}a$ to be the number of times a has appeared in $S^{/i}$, that is, $\#^{/i}a=\sum_{i'\neq i}\mathbb{1}_{[a\in z_{i'}]}$. For sequential event prediction, the left-hand side of a highest-scoring-correct rule for a general basket z on $S^{/i}$ obeys:

$$a^{+}_{S^{/i}ztK} \in \operatorname*{argmax}_{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\},a\in A} f_{S^{/i},K}(a,z_{\cdot,t+1}) = \operatorname*{argmax}_{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\},a\in A} \frac{\#^{/i}(a\cup z_{\cdot,t+1})}{\#^{/i}a+K}.$$

A highest-scoring-incorrect rule for basket z on $S^{/i}$ obeys:

$$[a^{-}_{S^{/i}ztK},b^{-}_{S^{/i}ztK}] \in \operatorname*{argmax}_{\substack{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\},a\in A\\ b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t+1}\}}} f_{S^{/i},K}(a,b) = \operatorname*{argmax}_{\substack{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\},a\in A\\ b\in\mathcal{X}\setminus\{z_{\cdot,1},\ldots,z_{\cdot,t+1}\}}} \frac{\#^{/i}(a\cup b)}{\#^{/i}a+K}.$$

In Lemma 16 below, we bound the difference in adjusted confidence of a general basket z when z_i is removed from the training set, in the sequential event prediction setting.

Lemma 16 *(Difference in Adjusted Confidence)*
Define $\tilde{a}_z := \min(\#a^-_{SztK}, \#^{/i}a^-_{S^{/i}ztK})$ *and* $\hat{a}_z := \min(\#a^+_{SztK}, \#^{/i}a^+_{S^{/i}ztK})$. *Then,*

$$\textit{(I)} \quad |f_{S,K}(a^-_{SztK}, b^-_{SztK}) - f_{S^{/i},K}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})| \leq \frac{1}{\tilde{a}_z + K}, \textit{ and}$$

$$\textit{(II)} \quad |f_{S,K}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S^{/i},K}(a^+_{S^{/i}ztK}, z_{\cdot,t+1})| \leq \frac{1}{\hat{a}_z + K}.$$

Proof Any itemset a is either in z_i or not, thus $\#^{/i}a \geq \#a - 1$ and $\#^{/i}a \leq \#a$. Also the number of times we see $a \cup b$ is less than or equal to the number of times we see a. These observations lead to the following inequalities that will be used throughout the proof:

$$\begin{aligned}
\#^{/i}(a^-_{SztK} \cup b^-_{SztK}) &\geq \#(a^-_{SztK} \cup b^-_{SztK}) - 1, && (10)\\
\#^{/i}a^-_{SztK} &\leq \#a^-_{SztK}, && (11)\\
\#(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK}) &\geq \#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK}), && (12)\\
\#a^-_{S^{/i}ztK} &\leq \#^{/i}a^-_{S^{/i}ztK} + 1, && (13)\\
\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK}) &\leq \#^{/i}a^-_{S^{/i}ztK}, && (14)\\
\#^{/i}(a^+_{SztK} \cup z_{\cdot,t+1}) &\geq \#(a^+_{SztK} \cup z_{\cdot,t+1}) - 1, && (15)\\
\#^{/i}a^+_{SztK} &\leq \#a^+_{SztK}, && (16)\\
\#(a^+_{S^{/i}ztK} \cup z_{\cdot,t+1}) &\geq \#^{/i}(a^+_{S^{/i}ztK} \cup z_{\cdot,t+1}), && (17)\\
\#a^+_{S^{/i}ztK} &\leq \#^{/i}a^+_{S^{/i}ztK} + 1, && (18)\\
\#^{/i}(a^+_{S^{/i}ztK} \cup z_{\cdot,t+1}) &\leq \#^{/i}a^+_{S^{/i}ztK}. && (19)
\end{aligned}$$

To prove (I) we provide upper bounds for both $f_{S,K}(a^-_{SztK}, b^-_{SztK}) - f_{S^{/i},K}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})$ and $f_{S^{/i},K}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK}) - f_{S,K}(a^-_{SztK}, b^-_{SztK})$. Using that for basket z the adjusted confidence of the highest-scoring-incorrect rule on $S^{/i}$, $a^-_{S^{/i}ztK} \to b^-_{S^{/i}ztK}$, exceeds that of another incorrect rule $a^-_{SztK} \to b^-_{SztK}$, and using inequalities (10) and (11),

$$\frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + K} \geq \frac{\#^{/i}(a^-_{SztK} \cup b^-_{SztK})}{\#^{/i}a^-_{SztK} + K} \geq \frac{\#(a^-_{SztK} \cup b^-_{SztK}) - 1}{\#a^-_{SztK} + K}.$$

Using the inequality above:

$$\begin{aligned}
& f_{S,K}(a^-_{SztK}, b^-_{SztK}) - f_{S^{/i},K}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK}) \\
&= \frac{\#(a^-_{SztK} \cup b^-_{SztK})}{\#a^-_{SztK} + K} - \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + K} \\
&\leq \frac{\#(a^-_{SztK} \cup b^-_{SztK})}{\#a^-_{SztK} + K} - \frac{\#(a^-_{SztK} \cup b^-_{SztK}) - 1}{\#a^-_{SztK} + K} = \frac{1}{\#a^-_{SztK} + K}. \qquad (20)
\end{aligned}$$

Considering the other direction, using that the highest-scoring-incorrect rule under S has higher adjusted confidence than the rule $a^-_{S^{/i}ztK} \to b^-_{S^{/i}ztK}$ and inequalities (12) and (13):

$$\frac{\#(a^-_{SztK} \cup b^-_{SztK})}{\#a^-_{SztK} + K} \geq \frac{\#(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#a^-_{S^{/i}ztK} + K} \geq \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + 1 + K}.$$

Using this, and inequality (14),

$$\begin{aligned}
& f_{S^{/i},K}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK}) - f_{S,K}(a^-_{SztK}, b^-_{SztK}) \\
&= \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + K} - \frac{\#(a^-_{SztK} \cup b^-_{SztK})}{\#a^-_{SztK} + K} \\
&\leq \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + K} - \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + 1 + K} \\
&= \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{(\#^{/i}a^-_{S^{/i}ztK} + K)(\#^{/i}a^-_{S^{/i}ztK} + 1 + K)} \\
&\leq \frac{\#^{/i}a^-_{S^{/i}ztK}}{(\#^{/i}a^-_{S^{/i}ztK} + K)(\#^{/i}a^-_{S^{/i}ztK} + 1 + K)} \leq \frac{1}{\#^{/i}a^-_{S^{/i}ztK} + K}.
\end{aligned}$$

Together with (20) this proves (I). The proof of part (II) is identical, using a^+_{SztK} and $a^+_{S^{/i}ztK}$ in the place of a^-_{SztK} and $a^-_{S^{/i}ztK}$, $z_{\cdot,t+1}$ in the place of b^-_{SztK} and $b^-_{S^{/i}ztK}$, and inequalities (15)-(19). ■

The following lemma is the backbone for our stability computations. The upper bound in this lemma depends only on the supports of the relevant rules. Recall that $\tilde{a}_z := \min(\#a^-_{SztK}, \#^{/i}a^-_{S^{/i}ztK})$ and $\hat{a}_z := \min(\#a^+_{SztK}, \#^{/i}a^+_{S^{/i}ztK})$.

Lemma 17 *(Large Support Implies Stability)*

$$\begin{aligned}
& |\ell_{\gamma,K_r}(f_{S,K}, z) - \ell_{\gamma,K_r}(f_{S^{/i},K}, z)| \\
&\leq \frac{1}{\gamma}\frac{1}{T_z}\sum_{t=0}^{T_z-1}\left[\frac{1}{\tilde{a}_z + K} + |K_r - K|\left[\frac{1}{\tilde{a}_z + K_r}\left(\frac{m}{m+K} + \frac{1}{\tilde{a}_z + K}\right)\right]\right. \\
&\quad \left. + \frac{1}{\hat{a}_z + K} + |K_r - K|\left[\frac{1}{\hat{a}_z + K_r}\left(\frac{m}{m+K} + \frac{1}{\hat{a}_z + K}\right)\right]\right].
\end{aligned}$$

Proof

$$\begin{aligned}
& |\ell_{\gamma,K_r}(f_{S,K}, z) - \ell_{\gamma,K_r}(f_{S^{/i},K}, z)| \\
&= \left|\frac{1}{T_z}\sum_{t=0}^{T_z-1} c_\gamma\left(f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^-_{SztK}, b^-_{SztK})\right)\right. \\
&\quad \left. - c_\gamma\left(f_{S^{/i},K_r}(a^+_{S^{/i}ztK}, z_{\cdot,t+1}) - f_{S^{/i},K_r}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})\right)\right| \\
&\leq \frac{1}{\gamma}\frac{1}{T_z}\sum_{t=0}^{T_z-1} |f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S,K_r}(a^-_{SztK}, b^-_{SztK}) \\
&\quad - f_{S^{/i},K_r}(a^+_{S^{/i}ztK}, z_{\cdot,t+1}) + f_{S^{/i},K_r}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})| \\
&\leq \frac{1}{\gamma}\frac{1}{T_z}\sum_{t=0}^{T_z-1} |f_{S,K_r}(a^-_{SztK}, b^-_{SztK}) - f_{S^{/i},K_r}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})| \\
&\quad + |f_{S,K_r}(a^+_{SztK}, z_{\cdot,t+1}) - f_{S^{/i},K_r}(a^+_{S^{/i}ztK}, z_{\cdot,t+1})| \\
&=: \frac{1}{\gamma}\frac{1}{T_z}\sum_{t=0}^{T_z-1} \text{term}_1 + \text{term}_2.
\end{aligned}$$

The first inequality above used that c_γ is $1/\gamma$-Lipschitz. Consider an upper bound for term_1 as follows from Lemma 15:

$$\begin{aligned}
\text{term}_1 &= |f_{S,K_r}(a^-_{SztK}, b^-_{SztK}) - f_{S^{/i},K_r}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})| \\
&\leq |f_{S,K}(a^-_{SztK}, b^-_{SztK}) - f_{S^{/i},K}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})| \left(1 + \frac{|K_r - K|}{\#a^-_{SztK} + K_r}\right) \\
&\quad + \frac{|K_r - K|}{\min(\#a^-_{SztK}, \#^{/i}a^-_{S^{/i}ztK}) + K_r} \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + K} \\
&\leq |f_{S,K}(a^-_{SztK}, b^-_{SztK}) - f_{S^{/i},K}(a^-_{S^{/i}ztK}, b^-_{S^{/i}ztK})| \left(1 + \frac{|K_r - K|}{\tilde{a}_z + K_r}\right) \\
&\quad + \frac{|K_r - K|}{\tilde{a}_z + K_r} \frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + K}.
\end{aligned}$$

Now incorporating Lemma 16 and that $\frac{\#^{/i}(a^-_{S^{/i}ztK} \cup b^-_{S^{/i}ztK})}{\#^{/i}a^-_{S^{/i}ztK} + K} \leq \frac{m-1}{m-1+K} \leq \frac{m}{m+K}$,

$$\begin{aligned}
\text{term}_1 &\leq \frac{1}{\tilde{a}_z + K} \left(1 + \frac{|K_r - K|}{\tilde{a}_z + K_r}\right) + \frac{|K_r - K|}{\tilde{a}_z + K_r} \frac{m}{m+K} \\
&= \frac{1}{\tilde{a}_z + K} + |K_r - K| \left[\frac{1}{\tilde{a}_z + K_r} \left(\frac{m}{m+K} + \frac{1}{\tilde{a}_z + K}\right)\right].
\end{aligned}$$

The same steps can be followed exactly for term_2. ■

The following lemma is used for the proof for the large sample bound.

Lemma 18 *(Asymptotic Expectation of $1/(\#a+K)$) For any itemset $a \in A$ and any $K \geq 0$,*

$$\mathbb{E}_{S\sim\mathcal{D}} \frac{1}{\#a + K} \leq \frac{1}{mp_a + K} + O\left(\frac{1}{m^2}\right),$$

where p_a is the probability that a random basket contains a, that is, $p_a = P_{z\sim\mathcal{D}}(a \subseteq z)$.

Since $\#a$ is binomially distributed, $\#a \sim \text{Binomial}(m, p_a)$, the proof of this lemma can be found by directly applying Lemma 21 in Appendix C.

We now give the proof of pointwise hypothesis stability for the large sample bound. We are interested in the change in adjusted confidence of specific basket z_i when that same basket is removed from the training set, that is on $S^{/i}$. Because Lemma 17 holds for any z, it also holds for z_i, where $\tilde{a}_{z_i} := \min(\#a^-_{Sz_itK}, \#^{/i}a^-_{S^{/i}z_itK})$ and $\hat{a}_{z_i} := \min(\#a^+_{Sz_itK}, \#^{/i}a^+_{S^{/i}z_itK})$.

Proof *(Of Theorem 7)* First, note that:

$$\begin{aligned}
\frac{1}{\tilde{a}_{z_i} + K_r} &= \frac{1}{\min(\#a^-_{Sz_itK}, \#^{/i}a^-_{S^{/i}z_itK}) + K_r} \\
&\leq \frac{1}{\min(\#^{/i}a^-_{Sz_itK}, \#^{/i}a^-_{S^{/i}z_itK}) + K_r} \leq \sum_{a\in\mathcal{A}} \frac{1}{\#^{/i}a + K_r}.
\end{aligned}$$

By the same reasoning, similar upper bounds hold for $1/(\tilde{a}_{z_i}+K)$, $1/(\hat{a}_{z_i}+K_r)$, and $1/(\hat{a}_{z_i}+K)$. Starting from Lemma 17 using specific basket z_i and incorporating these bounds on each fraction,

$$\begin{aligned}
&|\ell_{\gamma,K_r}(f_{S,K},z_i)-\ell_{\gamma,K_r}(f_{S^{/i},K},z_i)| \\
&\leq \frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\left[\sum_{a\in\mathcal{A}}\frac{1}{\#^{/i}a+K}+|K_r-K|\left[\left(\sum_{a\in\mathcal{A}}\frac{1}{\#^{/i}a+K_r}\right)\left(\frac{m}{m+K}+\sum_{a\in\mathcal{A}}\frac{1}{\#^{/i}a+K}\right)\right]\right]. \qquad (21)
\end{aligned}$$

We have also that for any K_r, using that $p_{\min A}\leq p_a$ for all $a\in\mathcal{A}$, and Lemma 18:

$$\mathbb{E}_{S^{/i}\sim\mathcal{D}^{m-1}}\sum_{a\in\mathcal{A}}\frac{1}{\#^{/i}a+K_r} \quad\leq\quad \frac{|\mathcal{A}|}{(m-1)p_{\min A}+K_r}+O\left(\frac{1}{m^2}\right). \qquad (22)$$

Thus from (21) and (22), for any $1\leq i\leq m$,

$$\begin{aligned}
&\mathbb{E}_{S\sim\mathcal{D}^m}|\ell_{\gamma,K_r}(f_{S,K},z_i)-\ell_{\gamma,K_r}(f_{S^{/i},K},z_i)| \\
&\leq \frac{2}{\gamma}\mathbb{E}_{z_i\sim\mathcal{D}}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\mathbb{E}_{S^{/i}\sim\mathcal{D}^{m-1}}\left[\sum_{a\in\mathcal{A}}\frac{1}{\#^{/i}a+K}\right. \\
&+ \left.|K_r-K|\left[\left(\sum_{a\in\mathcal{A}}\frac{1}{\#^{/i}a+K_r}\right)\left(\frac{m}{m+K}+\sum_{a\in\mathcal{A}}\frac{1}{\#^{/i}a+K}\right)\right]\right] \\
&\leq \frac{2}{\gamma}\mathbb{E}_{z_i\sim\mathcal{D}}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\frac{|\mathcal{A}|}{(m-1)p_{\min A}+K}+O\left(\frac{1}{m^2}\right) \\
&\quad +|K_r-K|\left[\left(\frac{|\mathcal{A}|}{(m-1)p_{\min A}+K_r}\right)\left(\frac{m}{m+K}\right)+O\left(\frac{1}{m^2}\right)\right] \\
&= \frac{2}{\gamma}\frac{|\mathcal{A}|}{(m-1)p_{\min A}+K}+|K_r-K|\frac{2}{\gamma}\left(\frac{|\mathcal{A}|}{(m-1)p_{\min A}+K_r}\right)\left(\frac{m}{m+K}\right)+O\left(\frac{1}{m^2}\right)=:\beta,
\end{aligned}$$

where in the second inequality, we moved the $(\sum_{a\in\mathcal{A}}1/(\#^{/i}a+K_r))(\sum_{a\in\mathcal{A}}1/(\#^{/i}a+K))$ terms into the $O\left(\frac{1}{m^2}\right)$. To see this, one can take a Taylor expansion around the mean for all of the terms similar to $\frac{1}{\#a+K}$ as follows:

$$\frac{1}{\#a+K}\approx\frac{1}{mp_a+K}-\frac{(\#a-mp_a)}{(mp_a+K)^2}+\frac{(\#a-mp_a)^2}{(mp_a+K)^3}+\ldots.$$

When these terms are multiplied together, the result is always $O\left(\frac{1}{m^2}\right)$. Thus, the algorithm has pointwise hypothesis stability β. Using β within the B&E theorem yields the result. ■

Proof *(Of Theorem 9)*
Starting from Lemma 17, we will use the minimum support threshold to provide the upper bound for the reciprocal of the support of rules. All of the steps used to derive Lemma 17 are valid for the max confidence, min support setting, only the notation needs to be changed. We define $\tilde{a}_{z,\theta}:=\min(\#a^-_{Sz\theta},\#^{/i}a^-_{S^{/i}z\theta})$, and now define also $\hat{a}_{z,\theta}:=\min(\#a^+_{Sz\theta},\#^{/i}a^+_{S^{/i}z\theta})$. Lemma 17 provides for $\bar{f}_{S,\theta}$ and using $K=0$:

$$\begin{aligned}
&|\ell_{\gamma,K_r}(\bar{f}_{S,\theta},z)-\ell_{\gamma,K_r}(\bar{f}_{S^{/i},\theta},z)| \\
&\leq \frac{1}{\gamma}\frac{1}{T_z}\sum_{t=0}^{T_z-1}\left[\frac{1}{\tilde{a}_{z,\theta}}+K_r\left[\frac{1}{\tilde{a}_{z,\theta}+K_r}\left(1+\frac{1}{\tilde{a}_{z,\theta}}\right)\right]+\frac{1}{\hat{a}_{z,\theta}}+K_r\left[\frac{1}{\hat{a}_{z,\theta}+K_r}\left(1+\frac{1}{\hat{a}_{z,\theta}}\right)\right]\right].
\end{aligned}$$

The requirement of a minimum support threshold ensures that for any particular item b, the highest scoring rule with b on the right must have support at least θ, that is: $\operatorname{argmax}_{a\subseteq\{z_{\cdot,1},\ldots,z_{\cdot,t}\},a\in A} \bar{f}_{S,\theta}(a,b)$ includes only itemsets with support at least θ. If b has never been ordered, $\max_a \bar{f}_{S,\theta}(a,b) = 0$ and we choose the maximizing rule to be $\varnothing \to b$, with support $m > m-1 \geq \theta$. By this reasoning, all of the rules we use have support at least θ: $\#a^{-}_{Sz t\theta} \geq \theta$, $\#^{/i}a^{-}_{S^{/i}z t\theta} \geq \theta$, $\#a^{+}_{Sz t\theta} \geq \theta$, and $\#^{/i}a^{+}_{S^{/i}z t\theta} \geq \theta$. Thus, $\tilde{a}_{z,\theta} \geq \theta$ and also $\hat{a}_{z,\theta} \geq \theta$. Using this in the previous expression:

$$
\begin{aligned}
&|\ell_{\gamma,K_r}(\bar{f}_{S,\theta},z) - \ell_{\gamma,K_r}(\bar{f}_{S^{/i},\theta},z)| \\
&\quad\leq \frac{2}{\gamma}\frac{1}{T_z}\sum_{t=0}^{T_z-1}\left[\frac{1}{\theta} + K_r\left[\left(\frac{1}{\theta+K_r}\right)\left(1+\frac{1}{\theta}\right)\right]\right] = \frac{2}{\gamma}\left[\frac{1}{\theta} + K_r\left(\frac{1}{\theta+K_r}\right)\left(1+\frac{1}{\theta}\right)\right] =: \beta.
\end{aligned}
$$

This expression holds for all S and for all z. It is thus an upper bound on the uniform stability. Using β within the B&E theorem yields the result. ■

The proofs of Theorems 7 and 9 for classification are essentially identical to those provided above for sequential event prediction. The left-hand side of a highest-scoring-correct rule for general basket x on $S^{/i}$ obeys:

$$
a^{+}_{S^{/i}xK} \in \operatorname*{argmax}_{a\subseteq x, a\in A} f_{S^{/i},K}(a,y) = \operatorname*{argmax}_{a\subseteq x, a\in A} \frac{\#^{/i}(a\cup y)}{\#^{/i}a + K}.
$$

And the left-hand side of a highest-scoring-incorrect rule for x on $S^{/i}$ obeys:

$$
a^{-}_{S^{/i}xK} \in \operatorname*{argmax}_{a\subseteq x, a\in A} f_{S^{/i},K}(a,-y) = \operatorname*{argmax}_{a\subseteq x, a\in A} \frac{\#^{/i}(a\cup -y)}{\#^{/i}a + K}.
$$

We further define $\tilde{a}_x = \min(\#a^{-}_{SxK}, \#^{/i}a^{-}_{S^{/i}xK})$ and $\hat{a}_x := \min(\#a^{+}_{SxK}, \#^{/i}a^{+}_{S^{/i}xK})$, and $\tilde{a}_{x_i}$ and $\hat{a}_{x_i}$ as the analogous quantities for specific basket x_i. Lemma 16, Lemma 17, and the proof of Theorem 7 all hold for classification by making the following substitutions in notation: $\tilde{a}_x$ and $\tilde{a}_{x_i}$ for $\tilde{a}_z$ and $\tilde{a}_{z_i}$; $\hat{a}_x$ and $\hat{a}_{x_i}$ for $\hat{a}_z$ and $\hat{a}_{z_i}$; a^{-}_{SxK} and $-y$ for $a^{-}_{Sz tK}$ and $b^{-}_{Sz tK}$; a^{+}_{SxK} and y for $a^{+}_{Sz tK}$ and $z_{\cdot,t+1}$; $a^{-}_{S^{/i}xK}$ and $-y$ for $a^{-}_{S^{/i}z tK}$ and $b^{-}_{S^{/i}z tK}$; $a^{+}_{S^{/i}xK}$ for $a^{+}_{S^{/i}z tK}$; $\ell^{\text{class}}_{\gamma,K_r}$ for ℓ_{γ,K_r}; and removing entirely $\frac{1}{T_z}\sum_{t=0}^{T_z-1}$. For Theorem 9, we again replace K with θ in the notation to define $\tilde{a}_{x,\theta} = \min(\#a^{-}_{Sx\theta}, \#^{/i}a^{-}_{S^{/i}x\theta})$ and $\hat{a}_{x,\theta} := \min(\#a^{+}_{Sx\theta}, \#^{/i}a^{+}_{S^{/i}x\theta})$, and then substitute $\tilde{a}_{x,\theta}$ and $\hat{a}_{x,\theta}$ for $\tilde{a}_{z,\theta}$ and $\hat{a}_{z,\theta}$ in the proof of the theorem.

The next lemma is specific to classification and is used for the small sample bound for the adjusted confidence algorithm.

Lemma 19 *(Support Thresholds for Adjusted Confidence, Classification)*
For specific basket x_i, it is true that:

$$\begin{aligned}
\frac{1}{\tilde{a}_{x_i}+K_r} &\leq \tilde{\alpha}_{K_r},\ \textit{where}\ \tilde{\alpha}_{K_r} = \frac{m+K-\#^{/i}(-y_i)}{K(\#^{/i}(-y_i))+K_r(m+K-\#^{/i}(-y_i))};\\
\frac{1}{\tilde{a}_{x_i}+K} &\leq \tilde{\alpha}_{K},\ \textit{where}\ \tilde{\alpha}_{K} = \frac{1}{K}\left(1-\frac{\#^{/i}(-y_i)}{m+K}\right);\\
\frac{1}{\hat{a}_{x_i}+K_r} &\leq \hat{\alpha}_{K_r},\ \textit{where}\ \hat{\alpha}_{K_r} = \frac{m+K-\#^{/i}y_i}{K(\#^{/i}y_i)+K_r(m+K-\#^{/i}y_i)};\ \textit{and,}\\
\frac{1}{\hat{a}_{x_i}+K} &\leq \hat{\alpha}_{K},\ \textit{where}\ \hat{\alpha}_{K} = \frac{1}{K}\left(1-\frac{\#^{/i}y_i}{m+K}\right).
\end{aligned}$$

Proof First we use the fact that on S, the adjusted confidence of the highest-scoring-incorrect rule for x_i, $a^{-}_{Sx_iK} \to -y_i$, exceeds that of the rule $\varnothing \to -y_i$:

$$\frac{\#a^{-}_{Sx_iK}}{\#a^{-}_{Sx_iK}+K} \geq \frac{\#(a^{-}_{Sx_iK}\cup -y_i)}{\#a^{-}_{Sx_iK}+K} \geq \frac{\#(-y_i)}{m+K} = \frac{\#^{/i}(-y_i)}{m+K},$$

where in the last step we used that basket x_i does not have label $-y_i$. Rearranging,

$$\#a^{-}_{Sx_iK} \geq \tilde{\sigma} \text{ where } \tilde{\sigma} := K\left(\frac{\#^{/i}(-y_i)}{m+K-\#^{/i}(-y_i)}\right).$$

Similarly, the adjusted confidence of the highest-scoring-incorrect rule for x_i with data set $S^{/i}$, $a^{-}_{S^{/i}x_iK} \to -y_i$, exceeds that of the rule $\varnothing \to -y_i$, thus:

$$\frac{\#^{/i}a^{-}_{S^{/i}x_iK}}{\#^{/i}a^{-}_{S^{/i}x_iK}+K} \geq \frac{\#^{/i}(a^{-}_{S^{/i}x_iK}\cup -y_i)}{\#^{/i}a^{-}_{S^{/i}x_iK}+K} \geq \frac{\#^{/i}(-y_i)}{m-1+K} \geq \frac{\#^{/i}(-y_i)}{m+K}.$$

Rearranging, we find that $\#^{/i}a^{-}_{S^{/i}x_iK} \geq \tilde{\sigma}$. Thus, $\tilde{a}_{x_i} = \min(\#a^{-}_{Sx_iK}, \#^{/i}a^{-}_{S^{/i}x_iK}) \geq \tilde{\sigma}$. We can derive a similar bound for $\hat{a}_{x_i}$, beginning with $\#a^{+}_{Sx_iK}$:

$$\frac{\#a^{+}_{Sx_iK}}{\#a^{+}_{Sx_iK}+K} \geq \frac{\#(a^{+}_{Sx_iK}\cup y_i)}{\#a^{+}_{Sx_iK}+K} \geq \frac{\#y_i}{m+K} = \frac{\#^{/i}y_i+1}{m+K} > \frac{\#^{/i}y_i}{m+K}.$$

The first equality uses that basket x_i has label y_i. Rearranging,

$$\#a^{+}_{Sx_iK} > \hat{\sigma} \text{ where } \hat{\sigma} := K\left(\frac{\#^{/i}y_i}{m+K-\#^{/i}y_i}\right).$$

Similarly for $\#^{/i}a^{+}_{S^{/i}x_iK}$:

$$\frac{\#^{/i}a^{+}_{S^{/i}x_iK}}{\#^{/i}a^{+}_{S^{/i}x_iK}+K} \geq \frac{\#^{/i}(a^{+}_{S^{/i}x_iK}\cup y_i)}{\#^{/i}a^{+}_{S^{/i}x_iK}+K} \geq \frac{\#^{/i}y_i}{m-1+K} \geq \frac{\#^{/i}y_i}{m+K}.$$

Rearranging, we find $\#^{/i}a^{-}_{S^{/i}x_iK} \geq \hat{\sigma}$. Thus $\hat{a}_{x_i} = \min(\#a^{+}_{Sx_iK}, \#^{/i}a^{+}_{S^{/i}x_iK}) \geq \hat{\sigma}$. These lower bounds on the supports are now used to create upper bounds for the reciprocals:

$$\frac{1}{\tilde{a}_{x_i}+K_r} \leq \frac{1}{\tilde{\sigma}+K_r} = \tilde{\alpha}_{K_r} \quad \text{and} \quad \frac{1}{\tilde{a}_{x_i}+K} \leq \frac{1}{\tilde{\sigma}+K} = \tilde{\alpha}_K.$$

The bounds for $\frac{1}{\hat{a}_{x_i}+K_r}$ and $\frac{1}{\hat{a}_{x_i}+K}$ are obtained in a similar way using $\hat{\sigma}$. ■

The proof of the small sample bound for classification follows directly from this lemma.

Proof *(Of Theorem 11)*
From Lemma 17, adapted for classification,

$$\begin{aligned}
&|\ell^{\text{class}}_{\gamma,K_r}(f_{S,K},z_i) - \ell^{\text{class}}_{\gamma,K_r}(f_{S^{/i},K},z_i)| \\
&\leq \frac{1}{\gamma}\left[\frac{1}{\tilde{a}_{x_i}+K} + |K_r-K|\left[\frac{1}{\tilde{a}_{x_i}+K_r}\left(\frac{m}{m+K}+\frac{1}{\tilde{a}_{x_i}+K}\right)\right]\right. \\
&\quad \left. + \frac{1}{\hat{a}_{x_i}+K} + |K_r-K|\left[\frac{1}{\hat{a}_{x_i}+K_r}\left(\frac{m}{m+K}+\frac{1}{\hat{a}_{x_i}+K}\right)\right]\right].
\end{aligned}$$

Combining this and Lemma 19, we have:

$$\begin{aligned}
&|\ell^{\text{class}}_{\gamma,K_r}(f_{S,K},z_i) - \ell^{\text{class}}_{\gamma,K_r}(f_{S^{/i},K},z_i)| \\
&\leq \frac{1}{\gamma}\left[\tilde{\alpha}_K + |K_r-K|\tilde{\alpha}_{K_r}\left(\frac{m}{m+K}+\tilde{\alpha}_K\right) + \hat{\alpha}_K + |K_r-K|\hat{\alpha}_{K_r}\left(\frac{m}{m+K}+\hat{\alpha}_K\right)\right] \\
&= \frac{1}{\gamma}(\tilde{\alpha}_K+\hat{\alpha}_K) + \frac{1}{\gamma}|K_r-K|\left[\tilde{\alpha}_{K_r}\left(\frac{m}{m+K}+\tilde{\alpha}_K\right) + \hat{\alpha}_{K_r}\left(\frac{m}{m+K}+\hat{\alpha}_K\right)\right].
\end{aligned}$$

We now provide an upper bound on the expectation of this quantity, beginning with the first term:

$$\begin{aligned}
\mathbb{E}_{z_1,\ldots,z_m}\frac{1}{\gamma}(\tilde{\alpha}_K+\hat{\alpha}_K) &= \mathbb{E}_{z_1,\ldots,z_m}\frac{1}{\gamma}\frac{1}{K}\left[\left(1-\frac{\#^{/i}(-y_i)}{m+K}\right)+\left(1-\frac{\#^{/i}y_i}{m+K}\right)\right] \\
&= \frac{1}{\gamma}\frac{1}{K}\left(2-\frac{(m-1)p_{-y_i}}{m+K}-\frac{(m-1)p_{y_i}}{m+K}\right) \\
&\leq \frac{2}{\gamma}\frac{1}{K}\left(1-\frac{(m-1)p_{y,\min}}{m+K}\right).
\end{aligned}$$

Here we used the fact that the mean of the binomial distribution $\text{Bin}(m-1,p_{y_i})$ is $(m-1)p_{y_i}$, and we use a lower bound for p_{y_i} and p_{-y_i}, namely $p_{y,\min} = \min(P(y=1),P(y=-1))$ the minimum

probability of a randomly chosen basket having any particular label. For the second term,

$$\begin{aligned}
&\mathbb{E}_{z_1,\ldots,z_m}\frac{1}{\gamma}|K_r-K|\left[\tilde{\alpha}_{K_r}\left(\frac{m}{m+K}+\tilde{\alpha}_K\right)+\hat{\alpha}_{K_r}\left(\frac{m}{m+K}+\hat{\alpha}_K\right)\right]\\
&= \frac{1}{\gamma}|K_r-K|\mathbb{E}_{z_1,\ldots,z_m}\left[\frac{1}{K\left(\frac{\#^{/i}(-y_i)}{m+K-\#^{/i}(-y_i)}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\#^{/i}(-y_i)}{m+K}\right)\right)\right]\\
&\quad +\frac{1}{\gamma}|K_r-K|\mathbb{E}_{z_1,\ldots,z_m}\left[\frac{1}{K\left(\frac{\#^{/i}y_i}{m+K-\#^{/i}y_i}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\#^{/i}y_i}{m+K}\right)\right)\right]\\
&= \frac{1}{\gamma}|K_r-K|\mathbb{E}_{\tilde{\zeta}\sim\mathrm{Bin}(m-1,p_{-y_i})}\left[\frac{1}{K\left(\frac{\tilde{\zeta}}{m+K-\tilde{\zeta}}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\tilde{\zeta}}{m+K}\right)\right)\right]\\
&\quad +\frac{1}{\gamma}|K_r-K|\mathbb{E}_{\hat{\zeta}\sim\mathrm{Bin}(m-1,p_{y_i})}\left[\frac{1}{K\left(\frac{\hat{\zeta}}{m+K-\hat{\zeta}}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\hat{\zeta}}{m+K}\right)\right)\right]\\
&=: \frac{1}{\gamma}|K_r-K|\mathbb{E}_{\tilde{\zeta}\sim\mathrm{Bin}(m-1,p_{-y_i})}F(\tilde{\zeta})+\frac{1}{\gamma}|K_r-K|\mathbb{E}_{\hat{\zeta}\sim\mathrm{Bin}(m-1,p_{y_i})}F(\hat{\zeta}).
\end{aligned}$$

Since the function $F(\zeta)$ is decreasing as ζ increases, then an upper bound is produced by using the distribution $\mathrm{Bin}(m-1,p_{y,\min})$:

$$\begin{aligned}
&\mathbb{E}_{z_1,\ldots,z_m}\frac{1}{\gamma}|K_r-K|\left[\tilde{\alpha}_{K_r}\left(\frac{m}{m+K}+\tilde{\alpha}_K\right)+\hat{\alpha}_{K_r}\left(\frac{m}{m+K}+\hat{\alpha}_K\right)\right]\\
&\leq \frac{2}{\gamma}|K_r-K|\mathbb{E}_{\zeta\sim\mathrm{Bin}(m-1,p_{y,\min})}F(\zeta)\\
&= \frac{2}{\gamma}|K_r-K|\mathbb{E}_{\zeta\sim\mathrm{Bin}(m-1,p_{y,\min})}\left[\frac{1}{K\left(\frac{\zeta}{m+K-\zeta}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\zeta}{m+K}\right)\right)\right].
\end{aligned}$$

■

The following lemma is similar to the previous lemma, but specific to sequential event prediction. It uses the support guarantee for the adjusted confidence algorithm (6) in order to bound the terms of Lemma 17, which holds with the same proof when the loss ℓ_{γ,K_r} is changed to the new loss $\ell^{\mathrm{new}}_{\gamma,K_r}$ and superscript "-" is replaced by "*". We define the analogy to $\tilde{a}_{z_i}$ as $\tilde{a}^*_{z_i} := \min(\#a^*_{Sz_itK},\#^{/i}a^*_{S^{/i}z_itK})$. The result below will immediately yield a proof of Theorem 12.

Lemma 20 *(Support Thresholds for Adjusted Confidence, Sequential Event Prediction)*
For specific basket z_i, define:

$$\alpha_{K_r} := \frac{m+K-\#z_{i,t+1}}{K(\#z_{i,t+1}-1)+K_r(m+K-\#z_{i,t+1})} \quad \textit{and} \quad \alpha_K := \frac{1}{K}\left(1-\frac{\#z_{i,t+1}-1}{m+K}\right).$$

It is true that:

$$\frac{1}{\tilde{a}^*_{z_i}+K_r}\leq\alpha_{K_r},\quad \frac{1}{\tilde{a}^*_{z_i}+K}\leq\alpha_K,\quad \frac{1}{\hat{a}_{z_i}+K_r}\leq\alpha_{K_r},\ \textit{and}\ \frac{1}{\hat{a}_{z_i}+K}\leq\alpha_K.$$

Proof Starting with (5), we know that $a^*_{Sz_itK} > \sigma$, where

$$\sigma := K\left(\frac{\#z_{i,t+1}-1}{m+K-\#z_{i,t+1}}\right).$$

We use the same type of argument as in (5), incorporating the fact that on $S^{/i}$, the adjusted confidence of the highest scoring rule $a^*_{S^{/i}z_itK} \to b^*_{S^{/i}z_itK}$ exceeds that of the highest-scoring-correct rule $a^+_{S^{/i}z_itK} \to z_{i,t+1}$, which exceeds that of the rule $\varnothing \to z_{i,t+1}$,

$$\begin{aligned}\frac{\#^{/i}a^*_{S^{/i}z_itK}}{\#^{/i}a^*_{S^{/i}z_itK}+K} &\geq \frac{\#^{/i}(a^*_{S^{/i}z_itK}\cup b^*_{S^{/i}z_itK})}{\#^{/i}a^*_{S^{/i}z_itK}+K} \geq \frac{\#^{/i}(a^+_{S^{/i}z_itK}\cup z_{i,t+1})}{\#^{/i}a^+_{S^{/i}z_itK}+K} \\ &\geq \frac{\#^{/i}z_{i,t+1}}{m-1+K} = \frac{\#z_{i,t+1}-1}{m-1+K}. \end{aligned} \tag{23}$$

Rearranging, we find that $\#^{/i}a^*_{S^{/i}z_itK} > \sigma$. Similarly for $\#a^+_{Sz_itK}$,

$$\frac{\#a^+_{Sz_itK}}{\#a^+_{Sz_itK}+K} \geq \frac{(\#a^+_{Sz_itK}\cup z_{i,t+1})}{\#a^+_{Sz_itK}+K} \geq \frac{\#z_{i,t+1}}{m+K}$$

so $\#a^+_{Sz_itK} \geq K\left(\frac{\#z_{i,t+1}}{m+K-\#z_{i,t+1}}\right) > \sigma$. And again for $\#^{/i}a^+_{S^{/i}z_itK}$ using (23),

$$\frac{\#^{/i}a^+_{S^{/i}z_itK}}{\#^{/i}a^+_{S^{/i}z_itK}+K} \geq \frac{\#^{/i}(a^+_{S^{/i}z_itK}\cup z_{i,t+1})}{\#^{/i}a^+_{S^{/i}z_itK}+K} \geq \frac{\#z_{i,t+1}-1}{m-1+K}.$$

so $\#^{/i}a^+_{S^{/i}z_itK} \geq \sigma$. We now have $\tilde{a}^*_{z_i} = \min(\#a^*_{Sz_itK}, \#^{/i}a^*_{S^{/i}z_itK}) \geq \sigma$, and also $\hat{a}_{z_i} = \min(\#a^+_{Sz_itK}, \#^{/i}a^+_{S^{/i}z_itK}) \geq \sigma$. Since σ is a lower bound on all the supports, it can be used to create an upper bound for the reciprocals, as follows, using $\tilde{a}^*_{z_i}$ as an example:

$$\frac{1}{\tilde{a}^*_{z_i}+K_r} \leq \frac{1}{\sigma+K_r} = \alpha_{K_r} \quad \text{and} \quad \frac{1}{\tilde{a}^*_{z_i}+K} \leq \frac{1}{\sigma+K} = \alpha_K.$$

■

Proof *(Of Theorem 12)* First, all of the steps in the proof of Lemma 17 hold when we replace the loss ℓ_{γ,K_r} with the new loss $\ell^{\text{new}}_{\gamma,K_r}$, replace c_γ with c^{new}_γ, and $\tilde{a}_{z_i}$ by $\tilde{a}^*_{z_i}$, so we obtain:

$$\begin{aligned}&|\ell^{\text{new}}_{\gamma,K_r}(f_{S,K},z_i) - \ell^{\text{new}}_{\gamma,K_r}(f_{S^{/i},K},z_i)| \\ &\leq \frac{1}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\left[\frac{1}{\tilde{a}^*_{z_i}+K} + |K_r-K|\left[\frac{1}{\tilde{a}^*_{z_i}+K_r}\left(\frac{m}{m+K}+\frac{1}{\tilde{a}^*_{z_i}+K}\right)\right]\right. \\ &\quad \left. + \frac{1}{\hat{a}_{z_i}+K} + |K_r-K|\left[\frac{1}{\hat{a}_{z_i}+K_r}\left(\frac{m}{m+K}+\frac{1}{\hat{a}_{z_i}+K}\right)\right]\right].\end{aligned}$$

Combining this and Lemma 20, we have:

$$|\ell^{\text{new}}_{\gamma,K_r}(f_{S,K},z_i) - \ell^{\text{new}}_{\gamma,K_r}(f_{S^{/i},K},z_i)| \leq \frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\alpha_K + |K_r-K|\alpha_{K_r}\left(\frac{m}{m+K}+\alpha_K\right).$$

To calculate the stability, we need an upper bound on the expectation of this quantity. Let us first create an upper bound for the expectation of the first term, $\frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\alpha_K$:

$$\begin{aligned}
\mathbb{E}_{z_1,\ldots,z_m}\frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\alpha_K &= \mathbb{E}_{z_1,\ldots,z_m}\frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\frac{1}{K}\left(1-\frac{\#z_{i,t+1}-1}{m+K}\right)\\
&= \mathbb{E}_{z_i}\frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\frac{1}{K}\left(1-\frac{\mathbb{E}_{z_1,\ldots,z_{i-1},z_{i+1},\ldots,z_m}\#z_{i,t+1}-1}{m+K}\right)\\
&= \mathbb{E}_{z_i}\frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\frac{1}{K}\left(1-\frac{(m-1)p_{z_{i,t+1}}}{m+K}\right)\\
&\leq \mathbb{E}_{z_i}\frac{2}{\gamma}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\frac{1}{K}\left(1-\frac{(m-1)p_{\min}}{m+K}\right)=\frac{2}{\gamma}\frac{1}{K}\left(1-\frac{(m-1)p_{\min}}{m+K}\right).
\end{aligned}$$

The first line above uses the definition of α_K, the second line uses the fact that each basket is chosen independently, the third line uses that $z_{i,t+1}$ is always contained in z_i and also uses the fact that the mean of the binomial distribution $\mathrm{Bin}(m-1,p_{z_{i,t+1}})$ is $(m-1)p_{z_{i,t+1}}$. The fourth line uses that $p_{z_{i,t+1}}$ has the lower bound $p_{\min}$, which no longer depends on z_i.

We repeat this outline for the second term:

$$\begin{aligned}
&\mathbb{E}_{z_1,\ldots,z_m}\frac{2}{\gamma}|K_r-K|\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\alpha_{K_r}\left(\frac{m}{m+K}+\alpha_K\right)\\
&= \mathbb{E}_{z_1,\ldots,z_m}\frac{2}{\gamma}|K_r-K|\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\frac{1}{K\left(\frac{\#z_{i,t+1}-1}{m+K-\#z_{i,t+1}}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\#z_{i,t+1}-1}{m+K}\right)\right)\\
&= \frac{2}{\gamma}|K_r-K|\mathbb{E}_{z_i}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\mathbb{E}_{z_1,\ldots,z_{i-1},z_{i+1},\ldots,z_m}\left[\frac{1}{K\left(\frac{\#z_{i,t+1}-1}{m+K-\#z_{i,t+1}}\right)+K_r}\times\right.\\
&\quad\left.\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\#z_{i,t+1}-1}{m+K}\right)\right)\right]\\
&= \frac{2}{\gamma}|K_r-K|\mathbb{E}_{z_i}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\mathbb{E}_{\zeta\sim\mathrm{Bin}(m-1,p_{z_{i,t+1}})}\frac{1}{K\left(\frac{\zeta+1-1}{m+K-\zeta-1}\right)+K_r}\left(\frac{m}{m+K}+\frac{1}{K}\left(1-\frac{\zeta+1-1}{m+K}\right)\right)\\
&=: \frac{2}{\gamma}|K_r-K|\mathbb{E}_{z_i}\frac{1}{T_{z_i}}\sum_{t=0}^{T_{z_i}-1}\mathbb{E}_{\zeta\sim\mathrm{Bin}(m-1,p_{z_{i,t+1}})}F(\zeta).
\end{aligned}$$

Algorithm 4: *Subroutine GenRules*, simplest version that considers only "marginal" rules.

Input: $(S,B,\mathcal{X})$, that is, past orders $S = \{z_i\}_{i=1,\ldots,m}$, $z_i \subseteq \mathcal{X}$, current basket $B \subset \mathcal{X}$, set of items $\mathcal{X}$

Output: Set of all rules where a_j is an item in the basket B (or the empty set) and b_j is not in B. That is, rules $\{a_j \to b_j\}_j$ such that $b_j \in \mathcal{X} \backslash B$ and either $a_j \in B$ or $a_j = \varnothing$.

Since the function F is decreasing as ζ increases, then an upper bound is produced by using the distribution $\mathrm{Bin}(m-1, p_{min})$. Namely,

$$\begin{aligned}
&\mathbb{E}_{z_1,\ldots,z_m} \frac{2}{\gamma} |K_r - K| \frac{1}{T_{z_i}} \sum_{t=0}^{T_{z_i}-1} \alpha_{K_r} \left(\frac{m}{m+K} + \alpha_K \right) \\
&\leq \frac{2}{\gamma} |K_r - K| \mathbb{E}_{z_i} \frac{1}{T_{z_i}} \sum_{t=0}^{T_{z_i}-1} \mathbb{E}_{\zeta \sim \mathrm{Bin}(m-1, p_{\min})} F(\zeta) \\
&= \frac{2}{\gamma} |K_r - K| \mathbb{E}_{\zeta \sim \mathrm{Bin}(m-1, p_{\min})} \frac{1}{K \left(\frac{\zeta}{m+K-\zeta-1} \right) + K_r} \left(\frac{m}{m+K} + \frac{1}{K} \left(1 - \frac{\zeta}{m+K} \right) \right).
\end{aligned}$$

■

In all of the theorems and proofs, the empirical loss and true loss are defined only for the case where the algorithm only recommends one item ($c = 1$). It is possible to use a vector norm to generalize to larger c.

6. Experiments

All data sets chosen for these experiments are publicly available from the UCI machine learning repository (Bache and Lichman, 2013), and from the IBM Quest Market-Basket Synthetic Data Generator (Agrawal and Srikant, 1994). To obtain formatted market-basket data, categorical data were converted into binary features (one feature per category). Each feature represents an item, and each example represents a basket. The feature value (0 or 1) indicates the presence of an item. Training baskets and test baskets were chosen randomly without replacement from the full data set. Since these data do not come naturally with a time ordering, items in the basket were randomly permuted to attain an order. At each iteration, rules were formed from one item or the empty item on the left, and one item on the right (See *GenRules* in Figure 4). Recommendations of one item were made using the following 15 algorithms: highest support, highest confidence, highest adjusted confidence for eight K levels, max confidence, min support algorithm for five support threshold levels θ. All 15 algorithms were evaluated by the average fraction of correct recommendations (AvgCorrect) per basket. As recommendations were made, it was common to have ties where multiple items were equally good to recommend, in which case the tie was broken at random; AvgCorrect is similar to $\ell_{0-1,K}$ except for this way of dealing with ties.

The parameters of the experiment are: number of training baskets (20 in all cases), number of test baskets (100 in all cases), values of K for the adjusted confidence algorithm (0.0001, 0.001, 0.01, 0.1, 1, 5, 10, 15), and values of θ for the max confidence, min support algorithm (1, 2, 3, 5,

10). Note that two of these algorithms are the same: the max confidence algorithm is the same as the max confidence, min support algorithm for θ=1. Data sets are: Car Evaluation (25 items, 1728 baskets), Chess King-Rook vs. King-Pawn, (75 items, 3196 baskets), MONK's problems (19 items, 1711 baskets) Mushroom (119 items, 8124 baskets), Nursery (32 items, 12960 baskets), Plants (70 items, 34781 baskets), T20I18D10KN22CR50 (22 items, 10000 baskets).

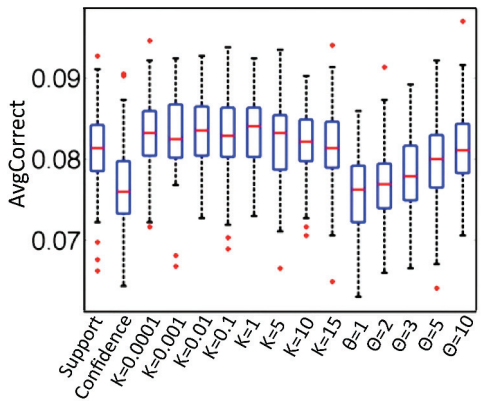
Each experiment (training, test, evaluation for all 15 algorithms) was performed 100 times, (totaling $100\times100\times15$ = 150,000 test basket evaluations per data set, for each of 7 data sets). In Figures 4 and 5, the distribution of AvgCorrect values for data sets Chess and Monk are shown via boxplot, along with the mean and standard deviation of AvgCorrect values. Bold indicates that the mean is not significantly different from that of the algorithm with the largest mean value; that is, bold indicates the highest scores. The boxplots and means for the other data sets are shown in Figures 7 through 11 in Appendix B.

Figure 6 summarizes the results of all of the experiments by totaling the number of data sets for which each algorithm achieved one of the highest scores. The best performing algorithms were $K = 0.01$ and $K = 0.1$, both algorithms achieving one of the top scores for 6 out of 7 of the data sets. The single data set for which these algorithms did not achieve one the best scores was the very dense data set T20I18D10KN22CR50, where the algorithms requiring a higher support (the max support algorithm, and also the adjusted confidence algorithm for $K = 5, 10$, and 15) achieved the highest AvgCorrect score. In that case, the $K = 0.01$ and $K = 0.1$ algorithms still performed better than the max confidence, min support algorithms for the parameters we tried.

The adjusted confidence algorithm with a very small K is similar to using the max confidence algorithm, except that whenever there is a tie, the tie is broken in favor of the rule with largest support. It seems that in most of the data sets we chose, this type of algorithm performed the best, which indicates two things. First, that for some data sets, increasing K too much can have the same effect as a too-large minimum support threshold, in that large values of K could potentially remove the best rules, leading to too much bias, and where the algorithm cannot explain enough of the variance in the data. Second, when comparing rules, it is important not to break ties at random as in the max confidence, min support algorithm, but instead to use the support of the rules. Another observation is that the performance levels of the adjusted confidence algorithm vary less than those of the max confidence, min support algorithm. In other words, our experiments indicate that a less-than-perfect choice of K for the adjusted confidence algorithm is likely to perform better than a less-than-perfect choice of θ for the max confidence, min support algorithm.

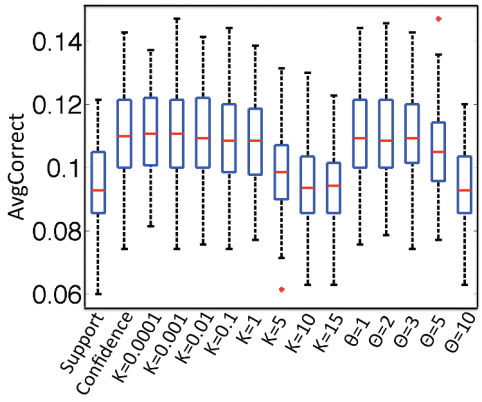
7. Related Work

We provide background on related works within several fields: association rule mining and associative classification, decision lists, recommender systems, and Bayesian analysis. There is also a body of literature on pattern mining in sequences, but not in the sequential event prediction setting defined here. This type of work generally considers the order in which items are added, and often uses a Markov assumption (see, for instance, Ayres et al., 2002; Berchtold and Raftery, 2002), whereas in our work, subsets of items are used to predict the next item, possibly without regard to the order in which they occurred, and a Markov assumption can be false. There is also work relating statistics to pattern mining and sequence mining, (e.g., Chernoff bounds for the confidence, Jacquemont et al., 2009). Our work also relates to multi-class classification, since there is a multi-class classification step at each point in time t of each sequence. For a recent work on generalization bounds in



Algorithm	mean ± standard dev.
Support	0.0813 ± 0.0046
Confidence	0.0764 ± 0.0053
K=0.0001	**0.0831 ± 0.0045**
K=0.001	**0.0832 ± 0.0048**
K=0.01	**0.0835 ± 0.0041**
K=0.1	**0.0831 ± 0.0049**
K=1	**0.0835 ± 0.0043**
K=5	0.0821 ± 0.0049
K=10	0.0821 ± 0.004
K=15	0.0816 ± 0.0049
θ=1	0.0759 ± 0.0049
θ=2	0.0767 ± 0.0045
θ=3	0.078 ± 0.0049
θ=5	0.0794 ± 0.0052
θ=10	0.0813 ± 0.0046

Figure 4: *Left*: Boxplots of AvgCorrect values for Chess data set. *Right*: Means and standard deviations for Chess data set.



Algorithm	mean ± standard dev.
Support	0.0943 ± 0.0126
Confidence	**0.1103 ± 0.0145**
K=0.0001	**0.1108 ± 0.0137**
K=0.001	**0.1109 ± 0.0147**
K=0.01	**0.1104 ± 0.0149**
K=0.1	**0.11 ± 0.0151**
K=1	0.1081 ± 0.0148
K=5	0.0992 ± 0.0138
K=10	0.0947 ± 0.0133
K=15	0.0948 ± 0.012
θ=1	**0.1098 ± 0.0138**
θ=2	**0.1095 ± 0.0146**
θ=3	**0.1092 ± 0.0146**
θ=5	0.1054 ± 0.0143
θ=10	0.0944 ± 0.0129

Figure 5: *Left:* Boxplots of AvgCorrect values for MONK's problems data set. *Right:* Means and standard deviations for MONK's problems data set.

multi-class classification see Shen and Wang (2007). Remember that in multi-class classification, each example is a feature vector, whereas in sequential event prediction, each example is an event sequence. Related work on generalization bounds includes those on algorithmic stability (Devroye and Wagner, 1979; Bousquet and Elisseeff, 2002).

7.1 Mining Association Rules

Association rule mining has proven successful for many applications, including market basket analysis (cross selling, product placement, affinity promotion, see also Kohavi et al., 2004), mining gene expression data (Jiang and Gruenwald, 2005), and weblog analysis (Huang et al., 2002). The ma-

Algorithm	Number of data sets
Support	1
Confidence	1
K=0.0001	4
K=0.001	5
K=0.01	6
K=0.1	6
K=1	2
K=5	2
K=10	2
K=15	2
θ=1	1
θ=2	1
θ=3	1
θ=5	0
θ=10	1

Figure 6: Summary of experiments: For each algorithm, the number of data sets where it performed comparably with the best algorithm.

jority of literature on association rule mining concerns the design of efficient algorithms to address the time-and-memory-consuming task of mining rules within very large databases. Discovering rules is usually a two-step process. First, itemsets are mined that meet a predetermined minimum support threshold. Then using this set, rules are formed and the strength of the rules is assessed using "interestingness" measures, such as the confidence. Many "interestingness" measures have been proposed in the literature (see Tan et al., 2002; Geng and Hamilton, 2007; McGarry, 2005). It is clearly possible to use the adjusted confidence as an interestingness measure for database exploration. In that setting, the adjusted confidence would provide a ranking of rules in terms of their ability to predict, including both "common sense rules" and "nuggets."

Although association rule mining has proven successful for many applications, it is well-known that the usefulness of association rules and their impact on even a wider range of practical applications remains limited due to problems arising from the minimum support threshold: first, the large number of rules mined can be intractable to domain experts who analyze rules and act on them, unless the minimum support threshold is set to a large value; second, the heuristic choice of the minimum support threshold tends to over-prune the search space of association rules, disregarding "nuggets" which can be very useful in many applications. Most prior work relies on the strong requirement of the minimum support threshold; some exceptions include the works of Li et al. (1999); Koh (2008) and DuMouchel and Pregibon (2001). Some recent work (Cohen et al., 2001; Wang et al., 2001) attempts to avoid the support measure altogether. In our work, the use of the adjusted confidence eliminates the need for the minimum support threshold.

When a set of rules is used to form a classifier, this is called "associative classification" (see, for instance, Liu et al., 1998; Thabtah, 2007; Vanhoof and Depaire, 2010).

7.2 Decision Lists

A decision list is an ordered set of association rules that forms a classifier (Rivest, 1987). Usually decision lists are formed the same way as decision trees are formed, which is by greedily splitting

on each nodes to form the tree, and then pruning (as in for instance, Li et al., 2001; Yin and Han, 2003; Simon et al., 2011; Marchand and Sokolova, 2005). However, it is possible to mine a set of rules, and order them to produce a classifier, as in the associative classification literature.

The work of Anthony (2004) contains a generalization bound for decision lists, but each rule in the list requires a linear combination, which is problematic in the sequential setting by the reasoning in Appendix A. (Similarly, there are many papers using a set of pre-computed rules as features for supervised learning, where a linear combination of rules is constructed, rather than a decision list; one recent example is by Friedman and Popescu 2008.)

In recent work, we have been learning the ordering of rules to form decision lists (Letham et al., 2013c).

7.3 Recommender Systems

Association rule mining has proven to be particularly useful for finding "goes with" relationships between items purchased simultaneously. Lin et al. (2002) also construct a recommender system using rules, having a minimum confidence threshold and then an adjustable minimum support threshold. Their scoring system is essentially based on support $\times$ confidence, which is not an estimate of $P(b|a)$ for rule $a \rightarrow b$. Lawrence et al. (2001) provide a recommender system for a grocery store, but the setting differs entirely from ours in that they always recommend items that have never been previously purchased.

In other work, we designed a Bayesian framework that estimates K for the adjusted confidence by "borrowing strength" across both users and items (McCormick et al., 2012). We are also looking at different approaches to the sequential event prediction problem, where we allow the predictions to alter the sequence in which items are placed into the basket (Letham et al., 2013b). This work uses a supervised learning framework for sequential event prediction.

We also note that a recommender system based on a weighted version of the adjusted confidence won third place in the ECML Discovery Challenge in 2013 (Letham, 2013).

Often, item-based collaborative filtering is used for problems that are actually sequential event prediction problems. There are several problems in applying standard item-based collaborative filtering techniques in sequential event prediction, the first one being that standard item-based collaborative filtering requires us to compute a similarity measure between all "co-rated" items. The similarity measure is often symmetric between two items, there is no distinguishing between $P(a|b)$ and $P(b|a)$. Even if item b is *always* found when a is found, $P(b|a) = 1$, is it possible for b not to be recommended when a is present, even with more than sufficient data to see the pattern. Further, for an incomplete basket, we do not have the ratings for all "co-rated" items, since there is no natural way to differentiate between items that have not yet been purchased in this transaction, and items that will not be purchased in this transaction, as both have a "rating" of 0 at time t. Thus, the only ratings that are available are ratings of "1" indicating that an item is in the basket. In other words, where the association rule approach we present here is intrinsically sequential, it is unnatural to force item-based collaborative filtering into a sequential framework. In general, item-based collaborative filtering is not based in a typical machine learning setting, in that it is not based on either loss minimization or probabilistic modeling (as the association rule approach is). The work of Letham et al. (2013b) also shows experimentally that item-based collaborative filtering can be worse than the max-confidence association rule approach.

7.4 Bayesian Analysis

DuMouchel and Pregibon (2001, "D&P") present a Bayesian approach to the identification of interesting itemsets. While not a rule mining algorithm per se, the approach could be extended to produce rules. D&P consider the ratio of observed itemset frequencies to baseline frequencies computed under a particular independence model. A prior distribution over the collection of such ratios results in shrinkage estimates for the true ratios. The amount of shrinkage depends on the observed frequency and tends to be more pronounced for less frequent itemsets. Our approach differs from D&P in several key regards. Most importantly we focus directly on Bayesian estimation for rules rather than itemsets. Second, D&P use an empirical Bayes approach to choose the prior hyperparameters. Since our approach requires just a single hyperparameter, K, we instead let the user choose an appropriate value (the value might be determined by cross validation or empirical Bayes). Finally, D&P perform a stratified analysis; one interesting future direction for our proposed approach would be to incorporate stratification.

Breese et al. (1998) present a number of different algorithms for collaborative filtering, including two Bayesian approaches. One of their Bayesian approaches clusters users while the other constructs a Bayesian network. Condliff et al. (1999) present a hierarchical Bayesian approach to collaborative filtering that "borrows strength" across users. Neither Breese et al. nor Condliff et al. focus on repeated purchases but both present ideas and techniques that may have relevance to future versions of our approach, especially the borrowing strength ideas.

Our recent work (McCormick et al., 2012; Letham et al., 2013c) uses Bayesian analysis to order rules into decision lists.

8. Conclusion

This work synthesizes tools from several fields to analyze the use of association rules in a new supervised learning framework. This analysis is necessarily different from that of classical supervised learning analysis; as we have discussed, association rules provide two mechanisms for generalization: first a large sample, and second, a minimum support of rules. We considered two simple algorithms based on association rules: a max confidence, min support algorithm, and the Bayesian adjusted confidence algorithm. Both algorithms have a parameter that creates a bound on the support, regulating a tradeoff between accuracy on the training set and generalization ability. We have also demonstrated that the adjusted confidence introduced here has some advantages over the minimum support threshold that is commonly considered in association rule mining: it allows rare rules to be used while still encouraging generalization, and among rules with similar confidence, it prefers those with larger support.

Acknowledgments

We would like to express thanks to Gene Kogan for helpful discussions and inspiration. Support for this project was provided by the National Science Foundation under grant IIS-1053407.

Appendix A. Regression and the Sequential Event Prediction Problem

By using association rules to model conditional probabilities for the sequential event prediction problem, we make a general assumption about the Markov chains governing our application, namely that a subset of knowledge about the current state can be used to predict the most likely future state. In this section we will address the suitability of two natural regression approaches that do not make this assumption. Let X_i be an indicator variable that is 1 if item i is in the current basket and 0 otherwise.

A.1 First Regression Method

Apply regression (e.g., logistic regression) to create a model for each item separately. Consider the model for the last item (item m), where the predictor variables will be X_i for $i \in \{1,\ldots,m-1\}$, and X_m will be the response variable. This model would provide:

$$P(X_m = 1 | X_1 = x_1, \ldots, X_{m-1} = x_{m-1}) = \frac{1}{1+\exp(f)},$$

where $f = \sum_{i=1}^{m-1} \lambda_i x_i + \lambda_{0,m}$, with each $x_i \in \{0,1\}$.

Because the data are being revealed sequentially, the correct application of this technique is not straightforward. Only a *partial* basket is available when predictions need to be made. It is incorrect to substitute the current state of the basket directly into the formula above. For instance, if the current basket contains items 1 and 2, so $X_1 = 1$ and $X_2 = 1$, it is incorrect to write $P(X_m | X_1 = 1, X_2 = 1) = \frac{1}{1+\exp(f)}$, where $f = \lambda_1 + \lambda_2 + \lambda_{0,m}$. This statement would be equivalent to the expression:

$$P(X_m = 1 | X_1 = 1, X_2 = 1) = P(X_m = 1 | X_1 = 1, X_2 = 1, X_3 = 0, \ldots, X_{m-1} = 0),$$

which is clearly false in general. It is not that, for instance, $X_3 = 0$, it is simply that X_3 is not yet realized.

On the other hand, it is possible to integrate in order to obtain conditional probability estimates:

$$P(X_m = 1 | X_1 = 1, X_2 = 1) = \sum_{x_3 = \{0,1\}, \ldots, x_{m-1} = \{0,1\}} P(X_m = 1 | X_1 = 1, X_2 = 1, X_3 = x_3 \ldots, X_{m-1} = x_m) \times P(X_3 = x_3, \ldots, X_{m-1} = x_m),$$

where estimates of $P(X_3 = x_3, \ldots, X_{m-1} = x_m)$ would need to be made also for every one of the 2^{m-3} combinations of $x_3, \ldots x_{m-1}$. Thus, this approach would rely on a large number of uncertain estimates (given limited data, and even moderately large m), each introducing errors into the final estimate. This is in contrast to the association rule approaches where a class of conditional probabilities are directly estimated. Further, the regression method provided above would not be able to be explained easily to customers or managers. In most circumstances, it would also require a large amount of computation between recommendations. Finally, it is not clear how to incorporate the order in which items are placed into the basket within this type of model, whereas it is trivial to incorporate this into the association rule techniques as discussed in Section 2.2.

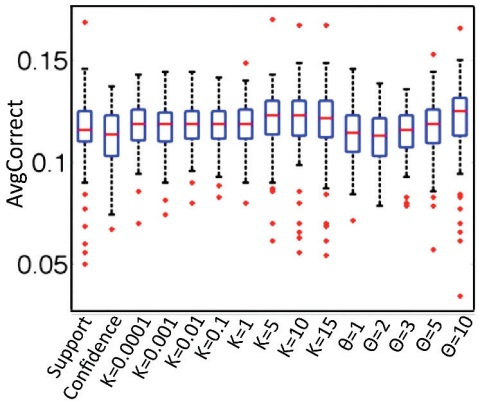
A.2 Second Regression Method

Apply regression methods (e.g., logistic regression) for each item and at each timestep, in total $m \times T$ regression models, where T is the size of the largest possible basket. This would give a direct way to incorporate time into the predictions. If the current basket contains t items, one would use only the models constructed using the first t items in each basket to predict the next item to be added. However, this would be making an entirely different assumption than the one given by the rule-mining approach. The rule-mining approach uses time only implicitly, and purchase patterns are counted the same regardless of the exact time within the transaction when the pattern occurred. In contrast, this regression approach would ignore all items added after time t in previous baskets. If apples were always followed by oranges, but in the past apples and oranges were always added after timestep t, then this approach would fail to recommend oranges when apples are added before timestep t. Further, the models for each timestep t must be constructed from baskets at least as large as t. This means that for very large baskets, there would only be a few past baskets that could be used to construct the models. Further, if the current basket is larger than any of the past baskets, the models would be trivial, since none of the past baskets can be used to construct them.

It may indeed be possible to use regression approaches for the sequential event prediction problem, but given the discussion above, it is not clear how this should be accomplished. We explore other ways to solve the sequential event prediction problem using supervised ranking techniques in another work (Letham et al., 2013b).

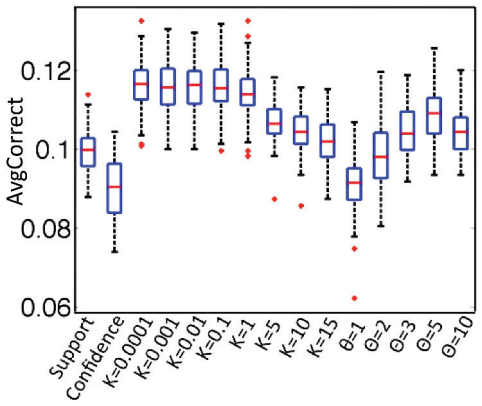
Appendix B. Additional Experimental Results

See Figures 7 - 11.



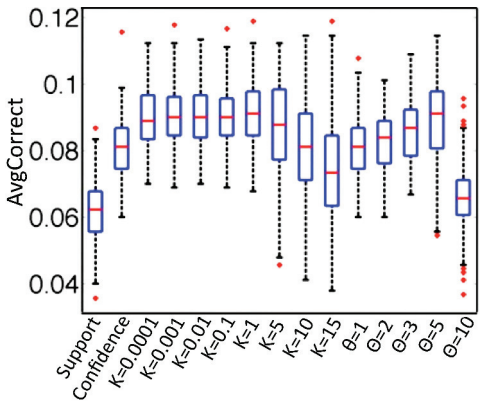
Algorithm	mean ± standard dev.
Support	0.115 ± 0.0176
Conf.	0.1125 ± 0.0143
K=0.0001	0.1173 ± 0.0127
K=0.001	0.1163 ± 0.0122
K=0.01	**0.1176 ± 0.0117**
K=0.1	**0.1177 ± 0.0109**
K=1	0.1176 ± 0.0116
K=5	**0.1204 ± 0.015**
K=10	**0.1199 ± 0.0172**
K=15	**0.1192 ± 0.0174**
θ=1	0.1133 ± 0.0134
θ=2	0.1119 ± 0.0131
θ=3	0.114 ± 0.0118
θ=5	0.1161 ± 0.0143
θ=10	**0.1205 ± 0.0191**

Figure 7: *Left*: Boxplots of AvgCorrect values for Cars data set. *Right*: Means and standard deviations for Cars data set.



Algorithm	mean ± standard dev.
Support	0.0996 ± 0.0051
Confidence	0.0902 ± 0.0075
K=0.0001	**0.1164 ± 0.0061**
K=0.001	**0.1158 ± 0.0062**
K=0.01	**0.1161 ± 0.0061**
K=0.1	**0.116 ± 0.0058**
K=1	0.1142 ± 0.0062
K=5	0.1069 ± 0.0052
K=10	0.1044 ± 0.0054
K=15	0.1024 ± 0.0053
θ=1	0.0909 ± 0.007
θ=2	0.0986 ± 0.0077
θ=3	0.1048 ± 0.0064
θ=5	0.1088 ± 0.0069
θ=10	0.1042 ± 0.0057

Figure 8: *Left:* Boxplots of AvgCorrect values for Mushroom data set. *Right:* Means and standard deviations for Mushroom data set.



Algorithm	mean ± standard dev.
Support	0.0619 ± 0.0098
Confidence	0.081 ± 0.0094
K=0.0001	0.0898 ± 0.0091
K=0.001	**0.0902 ± 0.0093**
K=0.01	**0.0902 ± 0.0085**
K=0.1	**0.0903 ± 0.0095**
K=1	**0.0909 ± 0.0096**
K=5	0.0869 ± 0.0139
K=10	0.0804 ± 0.0154
K=15	0.0747 ± 0.0154
θ=1	0.0811 ± 0.0088
θ=2	0.0819 ± 0.0094
θ=3	0.0858 ± 0.0095
θ=5	0.0883 ± 0.0137
θ=10	0.0654 ± 0.0111

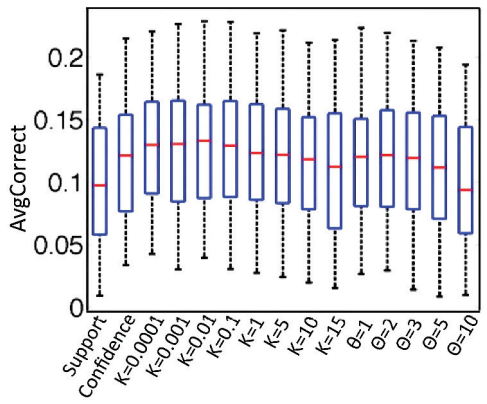
Figure 9: *Left:* Boxplots of AvgCorrect values for Nursery data set. *Right:* Means and standard deviations for Nursery data set.

Appendix C. Lemma 21

Lemma 21 *For $t \sim Binomial(m,p)$ and $K \geq 0$,*

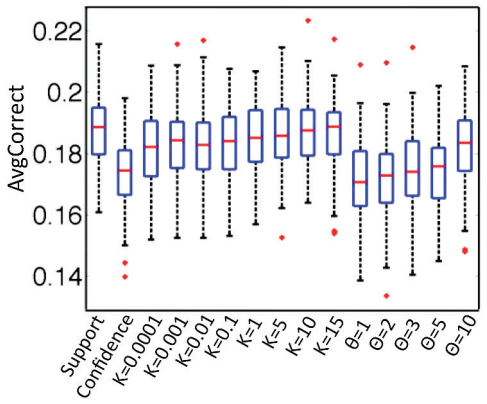
$$\mathbb{E}\left[\frac{1}{K+t}\right] = \frac{1}{K+mp} + O\left(\frac{1}{m^2}\right).$$

The proof of this lemma for $K = 0$ is provided by Rempala (2003). The proof of this lemma for $K > 0$ comes from Letham et al. (2013a), which we provide here for completeness. The proof of the lemma uses the following result.



Algorithm	mean ± standard dev.
Algorithm	mean pm standard dev.
Support	0.0983 ± 0.0494
Confidence	0.1187 ± 0.0465
K=0.0001	**0.1271 ± 0.0448**
K=0.001	**0.1251 ± 0.0454**
K=0.01	**0.1255 ± 0.0446**
K=0.1	**0.1251 ± 0.0464**
K=1	0.1235 ± 0.0454
K=5	0.1205 ± 0.0466
K=10	0.1141 ± 0.0464
K=15	0.1093 ± 0.0498
θ=1	0.1182 ± 0.0457
θ=2	0.1182 ± 0.0466
θ=3	0.118 ± 0.047
θ=5	0.11 ± 0.0511
θ=10	0.0981 ± 0.0496

Figure 10: *Left:* Boxplots of AvgCorrect values for Plants data set. *Right:* Means and standard deviations for Plants data set.



Algorithm	mean ± standard dev.
Support	**0.1874 ± 0.0115**
Confidence	0.1728 ± 0.0118
K=0.0001	0.1817 ± 0.012
K=0.001	0.1827 ± 0.0121
K=0.01	0.1821 ± 0.0124
K=0.1	0.183 ± 0.0125
K=1	0.1843 ± 0.0117
K=5	**0.1857 ± 0.0119**
K=10	**0.1871 ± 0.0115**
K=15	**0.1867 ± 0.0116**
θ=1	0.1722 ± 0.0126
θ=2	0.1716 ± 0.0128
θ=3	0.1748 ± 0.0131
θ=5	0.1742 ± 0.0125
θ=10	0.182 ± 0.0125

Figure 11: *Left:* Boxplots of AvgCorrect values for T20I18D10KN22CR50 data set. *Right:* Means and standard deviations for T20I18D10KN22CR50 data set.

Lemma 22 *Let* $X \sim \text{Binomial}(m,p)$ *and let* $\mu_k = \mathbb{E}\left[(X - \mathbb{E}[X])^k\right]$ *be the* k^{th} *central moment. For integer* $k \geq 1$, μ_{2k} *and* μ_{2k+1} *are* $O\left(m^k\right)$.

Proof We will use induction. For $k = 1$, the central moments are well known (e.g., Johnson et al., 2005): $\mu_2 = mp(1-p)$ and $\mu_3 = mp(1-p)(1-2p)$, which are both $O(m)$. We rely on the following recursion formula (Johnson et al., 2005; Romanovsky, 1923):

$$\mu_{s+1} = p(1-p)\left(\frac{d\mu_s}{dp} + ms\mu_{s-1}\right). \tag{24}$$

Because μ_2 and μ_3 are polynomials in p, their derivatives will also be polynomials in p. This recursion makes it clear that for all s, μ_s is a polynomial in p whose coefficients include terms involving m.

For the inductive step, suppose that the result holds for $k = s$. That is, μ_{2s} and μ_{2s+1} are $O(m^s)$. Then, by (24),

$$\mu_{2(s+1)} = p(1-p)\left(\frac{d\mu_{2s+1}}{dp} + (2s+1)m\mu_{2s}\right).$$

Differentiating μ_{2s+1} with respect to p yields a term that is $O(m^s)$. The term $(2s+1)m\mu_{2s}$ is $O(m^{s+1})$, and thus $\mu_{2(s+1)}$ is $O(m^{s+1})$. Also,

$$\mu_{2(s+1)+1} = p(1-p)\left(\frac{d\mu_{2(s+1)}}{dp} + 2(s+1)m\mu_{2s+1}\right).$$

Here $\frac{d\mu_{2(s+1)}}{dp}$ is $O(m^{s+1})$ and $2(s+1)m\mu_{2s+1}$ is $O(m^{s+1})$, and thus $\mu_{2(s+1)+1}$ is $O(m^{s+1})$.

This shows that if the result holds for $k = s$ then it must also hold for $k = s+1$ which completes the proof. ■

We can now prove Lemma 21.

Proof (Of Lemma 21) We expand $\frac{1}{K+X}$ at $X = mp$:

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{K+X}\right] &= \mathbb{E}\left[\sum_{i=0}^{\infty}(-1)^i\frac{(X-mp)^i}{(K+mp)^{i+1}}\right] \\
&= \sum_{i=0}^{\infty}(-1)^i\frac{\mathbb{E}\left[(X-mp)^i\right]}{(K+mp)^{i+1}} \\
&= \frac{1}{K+mp} + \sum_{i=2}^{\infty}(-1)^i\frac{\mu_i}{(K+mp)^{i+1}} \qquad (25)
\end{aligned}$$

where μ_i is the i^{th} central moment and we recognize that $\mu_1 = 0$. By Lemma 22,

$$\frac{\mu_i}{(K+mp)^{i+1}} = \frac{O\left(m^{\lfloor\frac{i}{2}\rfloor}\right)}{O\left(m^{i+1}\right)} = O\left(m^{\lfloor\frac{i}{2}\rfloor-i-1}\right).$$

The alternating sum in (25) can be split into two sums:

$$\sum_{i=2}^{\infty}(-1)^i\frac{\mu_i}{(K+mp)^{i+1}} = \sum_{i=2}^{\infty}O\left(m^{\lfloor\frac{i}{2}\rfloor-i-1}\right) = \sum_{i=2}^{\infty}O\left(\frac{1}{m^i}\right) + \sum_{i=3}^{\infty}O\left(\frac{1}{m^i}\right).$$

These are, for m large enough, bounded by a geometric series that converges to $O\left(\frac{1}{m^2}\right)$. ■

References

Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int'l Conf. Very Large Data Bases*, pages 487–499. Morgan Kaufmann, 1994.

Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Proc. ACM SIGMOD Int'l Conference on Management of Data*, pages 207–216, 1993.

Martin Anthony. Generalization error bounds for threshold decision lists. *Journal of Machine Learning Research*, 5:189–217, 2004.

Jay Ayres, Johannes Gehrke, Tomi Yiu, and Jason Flannick. Sequential PAttern Mining using a bitmap representation. In *Proceedings of the 8^{th} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

Kevin Bache and Moshe Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

André Berchtold and Adrian E. Raftery. The mixture transition distribution model for high-order markov chains and non-gaussian time series. *Statistical Science*, 17(3):328–356, 2002.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

John S. Breese, David Heckerman, and Carl Myers Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

Edith Cohen, Mayur Datar, Shinji Fujiwara, Aristides Gionis, Piotr Indyk, Rajeev Motwani, Jeffrey D. Ullman, and Cheng Yang. Finding interesting associations without support pruning. *IEEE Trans. Knowl. Data Eng.*, 13(1):64–78, 2001.

Michelle Keim Condliff, David D. Lewis, David Madigan, and Christian Posse. Bayesian mixed-effects models for recommender systems. In *ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

Luc P. Devroye and Terry J. Wagner. Distribution-free performance bounds for potential function rules. *IEEE Transactions on Information Theory*, 25(5):601–604, 1979.

William DuMouchel and Daryl Pregibon. Empirical bayes screening for multi-item associations. In *Proc. ACM SIGKDD Int'l Conf. on Knowl. Discovery and Data Mining*, pages 67–76, 2001.

Jerome H. Friedman and Bogdan E. Popescu. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3):916–954, 2008.

Liqiang Geng and Howard J. Hamilton. Choosing the right lens: Finding what is interesting in data mining. In *Quality Measures in Data Mining*, pages 3–24. Springer, 2007.

Xiangji Huang, Aijun An, Nick Cercone, and Gary Promhouse. Discovery of interesting association rules from Livelink web log data. In *Proc. IEEE Int'l Conf. on Data Mining*, 2002.

Peter Jackson. *Introduction to Expert Systems (Third Edition)*. Addison-Wesley, 1998.

Stéphanie Jacquemont, François Jacquenet, and Marc Sebban. A lower bound on the sample size needed to perform a significant frequent pattern mining task. *Pattern Recogn. Lett.*, 30:960–967, August 2009.

Xiang-Rong Jiang and Le Gruenwald. Microarray gene expression data association rules mining based on BSC-tree and FIS-tree. *Data & Knowl. Eng.*, 53(1):3–29, 2005.

Norman Lloyd Johnson, Adrienne W. Kemp, and Samuel Kotz. *Univariate Discrete Distributions*. John Wiley & Sons, August 2005.

Yun Sing Koh. Mining non-coincidental rules without a user defined support threshold. In *Advances in Knowl. Discovery and Data Mining, 12th Pacific-Asia Conf.*, pages 910–915, 2008.

Ron Kohavi, Llew Mason, Rajesh Parekh, and Zijian Zheng. Lessons and challenges from mining retail e-commerce data. *Machine Learning*, 57(1-2):83–113, 2004.

Richard D. Lawrence, George S. Almasi, Viveros Kotlyar, Marisa S. Viveros, and Sastry S. Duri. Personalization of supermarket product recommendations. *Data Mining and Knowledge Discovery*, 5(1-2):11–32, 2001.

Benjamin Letham. Similarity-weighted association rules for a name recommender system. In *Proceedings of the ECML/PKDD Discovery Challenge Workshop*, 2013.

Benjamin Letham, Cynthia Rudin, and Katherine A. Heller. Growing a list. *Data Mining and Knowledge Discovery*, 27(3):372–395, 2013a.

Benjamin Letham, Cynthia Rudin, and David Madigan. Sequential event prediction. *Machine Learning*, 93:357–380, 2013b.

Benjamin Letham, Cynthia Rudin, Tyler H. McCormick, and David Madigan. An interpretable stroke prediction model using rules and bayesian analysis. In *Proceedings of AAAI Late Breaking Track*, 2013c.

Jinyan Li, Xiuzhen Zhang, Guozho Dong, Kotagiri Ramamohanarao, and Qun Sun. Efficient mining of high confidence association rules without support thresholds. In *Proc. Principles of Data Mining and Knowledge Discovery*, pages 406–411, 1999.

Wenmin Li, Jiawei Han, and Jian Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. *IEEE International Conference on Data Mining*, pages 369–376, 2001.

Weiyang Lin, Sergio A. Alvarez, and Carolina Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data Mining and Knowledge Discovery*, 6(1):83–105, 2002.

Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, 1998.

David Madigan, Krzysztof Mosurski, and Russell G Almond. Graphical explanation in belief networks. *Journal of Computational and Graphical Statistics*, 6:160–181, 1997.

Mario Marchand and Marina Sokolova. Learning with decision lists of data-dependent features. *Journal of Machine Learning Research*, 2005.

Tyler H. McCormick, Cynthia Rudin, and David Madigan. Bayesian hierarchical modeling for predicting medical conditions. *The Annals of Applied Statistics*, 6(2):652–668, 2012.

Ken McGarry. A survey of interestingness measures for knowledge discovery. *The Knowledge Engineering Review*, 20:39–61, 2005.

Grzegorz A. Rempala. Asymptotic factorial powers expansions for binomial and negative binomial reciprocals. In *Proceedings of the American Mathematical Society*, volume 132, pages 261–272, 2003.

Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.

William H. Rogers and Terry J. Wagner. A finite sample distribution-free performance bound for local discrimination rules. *Annals of Statistics*, 6(3):506–514, 1978.

V. Romanovsky. Note on the moments of a binomial $(p+q)^n$ about its mean. *Biometrika*, 15: 410–412, 1923.

Cynthia Rudin, Benjamin Letham, Ansaf Salleb-Aouissi, Eugene Kogan, and David Madigan. A framework for supervised learning with association rules. In *Proceedings of the 24th Annual Conference on Learning Theory*, 2011.

Xiaotong Shen and Lifeng Wang. Generalization error for multi-class margin classification. *Electronic Journal of Statistics*, pages 307–330, 2007.

György J. Simon, Vipin Kumar, and Peter W. Li. A simple statistical model and association rule filtering for classification. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 823–831, 2011.

Pang N. Tan, Vipin Kumar, and Jaideep Srivastava. Selecting the right interestingness measure for association patterns. In *Proc. ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining*, 2002.

Fadi Thabtah. A review of associative classification mining. *The Knowledge Engineering Review*, 22:37–65, March 2007.

Koen Vanhoof and Benoît Depaire. Structure of association rule classifiers: a review. In *Proceedings of the International Conference on Intelligent Systems and Knowledge Engineering*, pages 9–12, 2010.

Vladimir Vapnik. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999, September 1999.

Ke Wang, Yu He, David Wai-Lok Cheung, and Francis Y. L. Chin. Mining confident rules without support requirement. In *Proc. Conference on Information and Knowledge Management*, pages 89–96, 2001.

Xiaoxin Yin and Jiawei Han. CPAR: Classification based on predictive association rules. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 331–335, 2003.

Journal of Machine Learning Research 14 (2013) 3493-3494 Submitted 12/12; Revised 8/13; Published 11/13

Comment on "Robustness and Regularization of Support Vector Machines" by H. Xu et al. (Journal of Machine Learning Research, vol. 10, pp. 1485-1510, 2009)

Yahya Forghani[*] YAHYAFOR2000@YAHOO.COM
Hadi Sadoghi H-SADOGHI@UM.AC.IR
Department of Computer
Ferdowsi University of Mashhad
Azadi Sq., Mashhad, IRAN

Editor: Ingo Steinwart

Abstract

This paper comments on the published work dealing with robustness and regularization of support vector machines (Journal of Machine Learning Research, Vol. 10, pp. 1485-1510, 2009) by H. Xu et al. They proposed a theorem to show that it is possible to relate robustness in the feature space and robustness in the sample space directly. In this paper, we propose a counter example that rejects their theorem.

Keywords: kernel, robustness, support vector machine

1. Comment

Firstly, it must be stated that Xu et al. (2009) made a good study of robustness and regularization of support vector machines. They proposed the following theorem to show that it is possible to relate robustness in the feature space and robustness in the sample space directly:

Theorem (Xu et al., 2009) *Suppose that the kernel function has the form* $k(x,x') = f(\|x-x'\|)$, *with* $f : R^+ \to R$ *a decreasing function. Denote by* H *the RKHS space of* $k(.,.)$ *and* $\phi(.)$ *the corresponding feature mapping. Then we have any* $x \in R^n, w \in H$ *and* $c > 0$,

$$\sup_{\|\delta\| \le c} \langle w, \phi(x-\delta)\rangle = \sup_{\|\delta_\phi\|_H \le \sqrt{2f(0)-2f(c)}} \left\langle w, \phi(x) - \delta_\phi \right\rangle .$$

The following counter example rejects the mentioned theorem. However, this theorem is a standalone result in the appendix of the paper of Xu et al. (2009), which is not used anywhere else in the paper of Xu et al. (2009). Thus, the main result and all other results of Xu et al. (2009) are not affected in any way.

Counter example. Let $\phi(.)$ be the feature mapping of Gaussian kernel function. We have $\|\phi(x)\|_H = 1$. Let $w = \phi(x)$. Therefore, $\langle w, \phi(x)\rangle = \|w\|_H$, and

$$\sup_{\|\delta\| \le c} \langle w, \phi(x-\delta)\rangle = \|w\|_H. \quad (1)$$

[*]. Also at Islamic Azad University, Mashhad Branch, Mashhad, IRAN.

Moreover,

$$\sup_{\|\delta_\phi\|_H \leq \sqrt{2f(0)-2f(c)}} \left\langle w, \phi(x) - \delta_\phi \right\rangle =$$

$$\sup_{\|\delta_\phi\|_H \leq \sqrt{2f(0)-2f(c)}} \langle w, \phi(x) \rangle + \sup_{\|\delta_\phi\|_H \leq \sqrt{2f(0)-2f(c)}} \left\langle w, \delta_\phi \right\rangle =$$

$$\|w\|_H + \sup_{\|\delta_\phi\|_H \leq \sqrt{2f(0)-2f(c)}} \left\langle w, \delta_\phi \right\rangle =$$

$$\|w\|_H + \|w\|_H \sqrt{2f(0) - 2f(c)}. \tag{2}$$

According to Equation (1) and (2), and since f is a decreasing function, for any $c > 0$, we have

$$\sup_{\|\delta\| \leq c} \langle w, \phi(x-\delta) \rangle \leq \sup_{\|\delta_\phi\|_H \leq \sqrt{2f(0)-2f(c)}} \left\langle w, \phi(x) - \delta_\phi \right\rangle.$$

End of counter example.

The exact spot that the error has been occurred in the mentioned theorem is Equation (19) of the paper of Xu et al. (2009). There it has been claimed that the image of the RKHS feature mapping is dense, which unfortunately is not true. Indeed, because $\langle \phi(x), \phi(x) \rangle = K(0)$ where $K(.)$ is the kernel function, the image of the feature mapping is in a ball of radius $\sqrt{K(0)}$.

References

Huan Xu, Shie Mannor, and Constantine Caramanis. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10:1485–1510, 2009.

Journal of Machine Learning Research 14 (2013) 3495-3536 Submitted 5/12; Revised 3/13; Published 12/13

Lovász ϑ function, SVMs and Finding Dense Subgraphs

Vinay Jethava JETHAVA@CHALMERS.SE
Anders Martinsson ANDEMAR@STUDENT.CHALMERS.SE
Computer Science & Engineering Department
Chalmers University of Technology
Rännvagen 6B
S 412 96, Göteborg, Sweden

Chiranjib Bhattacharyya CHIRU@CSA.IISC.ERNET.IN
Department of Computer Science and Automation
Indian Institute of Science
Bangalore, 560012
Karnataka, India

Devdatt Dubhashi DUBHASHI@CHALMERS.SE
Computer Science & Engineering Department
Chalmers University of Technology
Rännvagen 6B
S 412 96, Göteborg, Sweden

Editor: Francis Bach

Abstract

In this paper we establish that the Lovász ϑ function on a graph can be restated as a kernel learning problem. We introduce the notion of $\mathbf{SVM}-\vartheta$ graphs, on which Lovász ϑ function can be approximated well by a Support vector machine (SVM). We show that Erdös-Rényi random $G(n,p)$ graphs are $\mathbf{SVM}-\vartheta$ graphs for $\frac{\log^4 n}{n} \leq p < 1$. Even if we embed a large clique of size $\Theta\left(\sqrt{\frac{np}{1-p}}\right)$ in a $G(n,p)$ graph the resultant graph still remains a $\mathbf{SVM}-\vartheta$ graph. This immediately suggests an SVM based algorithm for recovering a large planted clique in random graphs. Associated with the ϑ function is the notion of orthogonal labellings. We introduce *common orthogonal labellings* which extends the idea of orthogonal labellings to multiple graphs. This allows us to propose a Multiple Kernel learning (MKL) based solution which is capable of identifying a large common dense subgraph in multiple graphs. Both in the planted clique case and common subgraph detection problem the proposed solutions beat the state of the art by an order of magnitude.

Keywords: orthogonal labellings of graphs, planted cliques, random graphs, common dense subgraph

1. Introduction

In a general graph many problems, such as computing the size of the largest clique or determining the chromatic number, are NP hard (Garey and Johnson, 1979). The ϑ function, introduced by Lovász (1979), is an extremely powerful tool for approximating such quantities in polynomial time. In some cases one can compute ϑ on large graphs efficiently, for example by exploiting symmetry (Bachoc et al., 2012), but in general efficient computation of ϑ function on large graphs remains a challenge. Evaluating the ϑ function on a graph requires solving a Semidefinite Program (SDP).

Using interior point methods one can solve SDPs, albeit with a high computational complexity of $O(n^6)$ (Boyd and Vandenberghe, 2004). Indeed numerical experiments show that computing ϑ on graphs consisting of more than 5000 vertices, using off the shelf SDP solvers, is impractical. Consider the problem of finding a large planted clique in a random graph. One could use an algorithm based on ϑ function computation (Feige and Krauthgamer, 2000) to recover such a clique. Unfortunately one cannot apply this algorithm to large graphs, say graphs of size more than $20,000$ vertices, due to high computational complexity of computing ϑ.

In this paper we establish that the ϑ function is equivalent to solving a kernel learning problem in the one class SVM setting. This surprising connection opens up lot of opportunities between graph theory and machine learning. Instead of trying to compute ϑ function exactly we show that by judicious choice of a kernel function, one can compute an upper-bound on the ϑ function by solving an SVM. We show that on random graphs this upper bound serves as a constant factor approximation to the ϑ function. We study how this bound can be exploited to identify large dense subgraphs in large graphs. In particular we study the problem of finding a *common dense subgraph* in multiple graphs (Pardalos and Rebennack, 2010), a computationally challenging problem for large graphs. We also study the problem of finding a hidden planted clique in a random graph. This is again an instance of computationally challenging problem (Jerrum, 1992).

1.1 The Importance of Studying Dense Subgraphs

Finding dense subgraphs in large graphs is an important problem, which has many applications in a variety of disciplines. In Computational Biology, mining for large dense subgraphs has important consequences for function discovery (Hu et al., 2005). Many other problems in Computational Biology can be posed as that of finding dense subgraphs (see, e.g., Spirin and Mirny, 2003; Jiang and Pei, 2009; Takahashi et al., 1987). In E-commerce one could find isolated submarkets, important for advertising, by finding dense subgraphs (Lang and Andersen, 2007). Many problems in social network analysis can also be posed as dense subgraph discovery problem (Newman et al., 2006). Dense subgraph discovery can also be useful in designing more secure systems (Applebaum et al., 2010). Recently a very interesting suggestion was made in Arora et al. (2010) where understanding the complexity of financial derivatives was linked to finding dense subgraphs. A comprehensive review of applications of dense subgraphs is beyond the scope of this paper and we refer the interested reader to the survey by Lee et al. (2010).

In this paper we target two difficult versions of dense subgraph recovery problem. The problem of planted clique in a random graph is an instance of dense subgraph discovery in a random graph. This problem is extensively studied by the Algorithms community. Though the focus of the study is mainly theoretical it also has practical implications in several disciplines including Machine Learning. To cite an example recently the problem of correlation detection was formulated as that of finding a planted clique in a large random graph (Devroye et al., 2011). Inspired by several applications in Computational Biology (see, e.g., Podolyan and Karypis, 2009), we study the problem of finding a large *common* dense subgraph in multiple graphs.

1.2 Contributions

In this paper we make several contributions. Lovász (1979) introduced the notion of orthogonal labellings and used it to define the ϑ function. We show that for any orthogonal labelling one can define a Kernel matrix, $\mathbf{K}$. Using this matrix $\mathbf{K}$, one can compute an upper-bound on the Lovász ϑ

function by solving a SVM. Furthermore we show that

$$\min_{\mathbf{K}\in\mathcal{K}(G)} \omega(\mathbf{K}) = \vartheta(G),$$

where $\omega(\mathbf{K})$ is the optimal SVM objective function. Finding a common dense region in multiple graphs is known to be computationally difficult problem. One of the main contribution of this paper is to show how the connection of ϑ to SVMs can be exploited to find a dense subgraph in multiple graphs. We extend the idea of orthogonal labelling to multiple graphs by introducing the notion of *common orthogonal labelling*. This allows us to use a formulation based on multiple kernel learning for this problem. The proposed method beats existing methods by an order of magnitude. Our results on the well-known benchmark DIMACS data set show that the proposed method can identify dense graphs in large variety of settings, while state of the art method fails. An important contribution of this paper is to introduce $\mathbf{SVM}-\vartheta$ graphs, on which Lovász ϑ function can be well approximated by SVM. It is interesting to note that $G(n,p)$ graphs are $\mathbf{SVM}-\vartheta$ graphs. In many approximation algorithms, the ϑ function needs to be computed on $G(n,p)$ graphs. An immediate consequence of our result is that one does not need to solve an SDP to compute the ϑ function but can potentially use an SVM to approximate it. An extremely difficult instance of dense subgraph recovery problem is to pose the question of finding a hidden clique in a random graph. State of the art approaches are not practical for large graphs as they use Lovász ϑ function (Feige and Krauthgamer, 2000). Another key contribution of this paper is show that one can find a planted clique by solving an SVM. In particular, we show that in a $G(n,1-p)$ graph even if we embed a clique of size $k=\Theta\big(\sqrt{n(1-p)/p}\big)$, the resultant graph is a $\mathbf{SVM}-\vartheta$ graph. Furthermore, even if embed a sparse *random* subgraph in a large random graph, the resultant graph turns out to be $\mathbf{SVM}-\vartheta$ graph. In both cases one can prove that the SVM solution can be used to recover the planted subgraph.

1.2.1 STRUCTURE OF THE PAPER

In Section 2 we review the definition of orthogonal labeling and establish a connection between Lovász ϑ function and a kernel learning problem on one class problem. We extend the notion of orthogonal labellings for single graphs to include multiple graphs. We introduce the notion of common orthogonal labeling to multiple graphs in Section 3. This leads to a MKL based formulation which is capable of finding a common dense region. Next we present one of the major contributions of this paper. In Section 4 we establish that there exists graphs on which the Lovász ϑ function can be well approximated by an SVM. Furthermore we show that the the graph associated with the planted clique problem also satisfies this property. In Section 5 we empirically evaluate the performance of the proposed algorithms on large variety of graphs.

1.2.2 NOTATION

We represent vectors using lower case bold letters $\mathbf{a},\mathbf{b},\ldots$, etc., and matrices using upper case bold letters $\mathbf{A},\mathbf{B},\ldots$ etc.; with a_i referring to i^{th} element of $\mathbf{a}$, and similarly A_{ij} referring to $(i,j)^{th}$ entry of matrix $\mathbf{A}$. We use notation $[n]$ to denote the set $\{1,2,\ldots,n\}$. For a vector in $\mathbb{R}^d$, we denote the Euclidean norm by $\|.\|$ and the infinity norm by $\|.\|_\infty$. The inequality $\mathbf{a}\geq 0$ is true if it the inequality holds element-wise. Let $\mathcal{S}^{d-1}=\{\mathbf{u}\in\mathbb{R}^{d-1}\,|\,\|\mathbf{u}\|_2=1\}$ denote a $(d-1)$ dimensional sphere. Let $\mathbf{S}_n$ denote the set of $n\times n$ square symmetric matrices and $\mathbf{S}_n^+$ denote $n\times n$ square symmetric positive semidefinite matrices. For any matrix $\mathbf{A}\in\mathbb{R}^{n\times n}$, we denote the eigenvalues $\lambda_1(\mathbf{A})\geq\ldots\geq\lambda_n(\mathbf{A})$,

and $\|\mathbf{A}\| = \sqrt{\lambda_1(\mathbf{A}^\top \mathbf{A})}$. $\texttt{diag}(\mathbf{r})$ will denote a diagonal matrix with diagonal entries defined by components of $\mathbf{r}$.

Support vector machines (SVMs) have emerged as a powerful tool for binary classification problems (Vapnik, 1995). SVMs are posed as a Convex Quadratic program (CQP) and can be solved in linear time (Hush et al., 2006). In this paper we will extensively use a variation of the SVM formulation, known as the one-class SVM (Schölkopf et al., 2001), and written as

$$\omega(\mathbf{K}) = \max_{\alpha_i \geq 0, i=1,\ldots,n} f(\alpha; \mathbf{K}) \left(= 2\sum_{i=1}^{n} \alpha_i - \sum_{i,j=1}^{n} \alpha_i \alpha_j K_{ij} \right), \tag{1}$$

where $\mathbf{K} \in \mathbf{S}_n^+$ is called the kernel matrix. This formulation can be solved in $O(n^2)$ time (Hush et al., 2006). In the sequel, for the sake of brevity we will denote (1) as SVM formulation.

Let $G = (V, E)$ be a graph of order n with vertex set $V = [n]$ and edge set $E \subseteq V \times V$. Let $\mathbf{A} \in \mathbf{S}_n$ denote the adjacency matrix of G where $A_{ij} = 1$ if edge $(i, j) \in E$, and 0 otherwise. Let $\bar{G}$ denote the complement graph of G. The adjacency matrix of $\bar{G}$ is $\bar{\mathbf{A}} = \mathbf{e}\mathbf{e}^\top - \mathbf{I} - \mathbf{A}$, where $\mathbf{e} = [1, 1, \ldots, 1]^\top$ is a vector of length n containing all 1's, and $\mathbf{I}$ denotes the identity matrix. We denote the indicator vector for some set $S \subseteq V$ as $\mathbf{e}_S$ which is one for all $i \in S$ and zero in other co-ordinates.

Let $N_i(G) = \{j : (i, j) \in E\}$ denote the neighbourhood of node $i \in V$; $d_i(G) = |N_i(G)|$ denote the degree of node i; and $\gamma(G) = |E| / \binom{|V|}{2}$ denote the density of graph G. Let $G_S = (S, E_S)$ denote the subgraph induced by $S \subseteq V$ in graph G. An independent set in G (a clique in $\bar{G}$) is a subset of vertices $S \subseteq V$ for which no (every) pair of vertices has an edge in G (in $\bar{G}$). Our notations are standard, see Bollobás (1998).

An event $\mathcal{A}_n$ holds with high probability if $P(\mathcal{A}_n)$ tends to 1 as n goes to ∞. The notations O, o, Ω, Θ will denote the standard measures defined in asymptotic analysis (see, e.g., Cormen et al., 2009, Chapter 3).

2. Lovász theta Function and Kernel Learning

Consider the problem of embedding a graph $G = (V, E)$ on an unit sphere S^{d-1}. The study of this problem was initiated by Lovász (1979) who introduced the idea of *orthogonal labelling*:

Definition 1 (Lovász, 1979) *An orthogonal labelling of graph $G = (V, E)$ with $|V| = n$, is a matrix $U = [\mathbf{u}_1, \ldots, \mathbf{u}_n] \in \mathbb{R}^{d \times n}$ such that $\mathbf{u}_i^\top \mathbf{u}_j = 0$ whenever $(i, j) \notin E$ and $\mathbf{u}_i \in S^{d-1}$ $\forall\, i \in [n]$.*

Let $Lab(G)$ denote the set of orthogonal labellings of graph G, given by:

$$Lab(G) := \{\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_n] \,|\, \mathbf{u}_i \in S^{d-1},\ \mathbf{u}_i^\top \mathbf{u}_j = 0 \,\forall\, (i, j) \notin E\}.$$

Using $Lab(G)$, Lovász (1979) defined $\vartheta(G)$ as follows:

$$\vartheta(G) = \min_{\mathbf{U} \in Lab(G)} \min_{\mathbf{c} \in S^{d-1}} \max_i \left(\mathbf{c}^\top \mathbf{u}_i\right)^{-2}.$$

In the sequel we will sometimes denote $\vartheta(G)$ by ϑ when the argument is clear from the context. There exist several other equivalent definitions of ϑ function, for a comprehensive discussion see monograph by Knuth (1994).

It can be shown that ϑ serves as an upper-bound on the size of maximum independent set, $\mathbf{ALPHA}(G)$,[1] of a graph G (Lovász, 1979). Indeed for any graph G,

$$\mathbf{ALPHA}(G) \leq \vartheta(G).$$

Computing $\mathbf{ALPHA}(G)$ is a classic NP-hard problem (Garey and Johnson, 1979), which is furthermore known to be very hard even to approximate (Håstad, 1999). However $\vartheta(G)$, which can be computed in polynomial time by solving an SDP, gives a polynomial time computable upper-bound on $\mathbf{ALPHA}(G)$. Since then Lovász ϑ function has been extensively used in solving a variety of algorithmic problems (see, e.g., Coja-Oghlan and Taraz, 2004; Krivelevich, 2002; Karger et al., 1998).

2.1 The Relationship between SVMs and $\vartheta(G)$

In this subsection we establish that the $\vartheta(G)$ function can be re-stated as a Kernel learning problem. An interesting characterization of ϑ function involving convex quadratic program (CQP) was given by Luz and Schrijver (2006), which we describe below:

Theorem 2 (Luz and Schrijver, 2006) *For a graph $G = (V,E)$ having n vertices, let $\mathbf{C} \in \mathbb{R}^{n\times n}$ be an $n \times n$ matrix with $C_{ij} = 0$ whenever $(i,j) \notin E$. Then,*

$$\begin{aligned} \vartheta(G) &= \min_{\mathbf{C}} \quad v(G,\mathbf{C}), \text{ where} \\ v(G,\mathbf{C}) &= \max_{\mathbf{x}\geq 0} \underbrace{2\mathbf{x}^\top \mathbf{e} - \mathbf{x}^\top \left(\frac{\mathbf{C}}{-\lambda_n(\mathbf{C})} + \mathbf{I}\right)\mathbf{x}}_{g_G(\mathbf{x})}. \end{aligned}$$

Proof See Luz and Schrijver (2006). ■

The above theorem can also be understood from a Kernel learning perspective in the SVM setting. Observe that for every feasible choice of $\mathbf{C}$, there exists an orthogonal labelling, $\mathbf{U}$ where $\mathbf{U}^\top\mathbf{U} = \mathbf{I} + \frac{\mathbf{C}}{-\lambda_n(\mathbf{C})}$. Taking a cue from this observation we state and prove the following theorem from first principles.

Theorem 3 *For an undirected graph $G = (V,E)$, with $|V| = n$, let*

$$\mathcal{K}(G) := \{\mathbf{K} \in \mathbf{S}_n^+ \mid K_{ii} = 1, i \in [n], K_{ij} = 0, (i,j) \notin E\}.$$

Then,

$$\vartheta(G) = \min_{\mathbf{K}\in\mathcal{K}(G)} \omega(\mathbf{K}).$$

Proof We begin by noting that any $\mathbf{K} \in \mathcal{K}(G)$ is positive semidefinite and hence there exists $\mathbf{U} \in \mathbb{R}^{d\times n}$ such that $\mathbf{K} = \mathbf{U}^\top\mathbf{U}$. Note that $K_{ij} = \mathbf{u}_i^\top \mathbf{u}_j$ where $\mathbf{u}_i$ is a column of $\mathbf{U}$. Hence by inspection it is clear that the columns of $\mathbf{U}$ define an orthogonal labelling on G, that is, $bU \in Lab(G)$. Using a similar argument we can show that for any $\mathbf{U} \in Lab(G)$, the matrix $\mathbf{K} = \mathbf{U}^\top\mathbf{U}$, is an element of $\mathcal{K}(G)$. The set of valid kernel matrices $\mathcal{K}(G)$ is thus equivalent to $Lab(G)$. Note that if $\mathbf{U}$

1. Usually in Algorithms literature, the size of the largest independent set in a graph G is denoted by $\alpha(G)$. We have chosen to denote it by $\mathbf{ALPHA}(G)$ to avoid conflict with the notation α_i for Support vectors.

is a labelling then $\mathbf{U} = \mathbf{U}\texttt{diag}(\varepsilon)$ is also an orthogonal labelling for any $\varepsilon^\top = [\varepsilon_1, \ldots, \varepsilon_n]$ where $\varepsilon_i \in \{1, -1\} \forall\, i \in [n]$. It thus suffices to consider only those labellings for which $\mathbf{c}^\top \mathbf{u}_i \geq 0 \forall\, i \in [n]$ holds. For a fixed $\mathbf{c}$, one can rewrite

$$\max_i \frac{1}{(\mathbf{c}^\top \mathbf{u}_i)^2} = \left(\min_t t^2 \text{ s.t. } \frac{1}{\mathbf{c}^\top \mathbf{u}_i} \leq t \; \forall\, i \in [n] \right).$$

Setting $\mathbf{w} = 2t\mathbf{c}$ yields the following relation

$$\min_{\mathbf{c} \in \mathcal{S}^{d-1}} \max_i \frac{1}{(\mathbf{c}^\top \mathbf{u}_i)^2} = \left(\min_{\mathbf{w} \in \mathbb{R}^d} \frac{\|\mathbf{w}\|^2}{4} \text{ s.t. } \mathbf{w}^\top \mathbf{u}_i \geq 2 \; \forall\, i \in [n] \right).$$

This establishes that given an orthogonal labelling $\mathbf{U}$, the minimization of $\mathbf{c}$ is obtained by solving a convex quadratic program (CQP) which is equivalent to solving a SVM. Application of strong duality immediately leads to the claim

$$\min_{\mathbf{c} \in \mathcal{S}^{d-1}} \max_i \frac{1}{(\mathbf{c}^\top \mathbf{u}_i)^2} = \omega(\mathbf{K})$$

where $\mathbf{K} = \mathbf{U}^\top \mathbf{U}$ and $\omega(\mathbf{K})$ is defined in (1). As there is a correspondence between each element of $Lab(G)$ and $\mathcal{K}$ minimization of $\omega(\mathbf{K})$ over $\mathcal{K}$ is equivalent to computing the $\vartheta(G)$ function. ■

Theorem 3 establishes a connection between $\vartheta(G)$ and $\omega(\mathbf{K})$, two well studied formulations in Graph theory and Machine Learning. As a consequence this immediately opens up the possibility of applying large-scale Kernel learning algorithms (see, e.g., Hu et al., 2011) for computing ϑ function. In this paper we do not explore this direction further, leaving it for future study, but instead focus on another important consequence of Theorem 3. The theorem establishes that if we choose not to optimize over all possible labellings but instead fix an orthogonal labelling and then, one can easily compute an upper-bound on $\mathbf{ALPHA}(G)$ by solving an SVM. See that for any graph G

$$\mathbf{ALPHA}(G) \leq \vartheta(G) \leq \omega(\mathbf{K}) \; \forall\, \mathbf{K} \in \mathcal{K}(G).$$

As SVMs have linear time complexity (Hush et al., 2006) this could be an efficient alternative for computing upper-bounds on size of the maximum independent set for any chosen labelling.

In Theorem 2, if we fix $\mathbf{C} = \mathbf{A}$, the adjacency matrix, we obtain a very interesting orthogonal labelling, which we will refer to as **LS** labelling as it was first introduced by Luz and Schrijver (2006). In Q graphs, introduced by Luz (1995), the **LS** labelling recovers $\mathbf{ALPHA}(G)$ by solving a CQP. The CQP is obtained by setting $\mathbf{C} = \mathbf{A}$ in Theorem 2.

Definition 4 (Luz, 1995) *A graph G, with adjacency matrix $\mathbf{A}$, is called a Q graph whenever $\mathbf{ALPHA}(G) = v(G, \mathbf{A})$ where $v(G, \mathbf{A})$ is defined in Theorem 2.*

Indeed, on a Q graph, computation of the ϑ function reduces to computing the minimum eigenvalue and solving a CQP, which in general has a complexity of $O(n^3)$. By Theorem 3, one can now state that on Q graphs evaluating the ϑ function is equal to solving an SVM, a special case of CQP, which can be solved in $O(n^2)$ time (Hush et al., 2006). Keeping computational advantages in mind it is interesting to characterize Q graphs.

Theorem 5 (Luz, 1995) *For a simple unweighted graph $G = (V,E)$, let $\mathbf{A}$ denote the adjacency matrix , S be the largest independent set, $v(G,\mathbf{A})$ and $g_G(x)$ be defined as in Theorem 2. The graph G is a Q graph iff*

$$-\lambda_n(A) \leq \min_{i \in V \setminus S} |N_i(G) \cap S|.$$

Furthermore $v(G,A) = g_G(\mathbf{e}_S)$, where $\mathbf{e}_S$ is the indicator vector on S.

Proof See Theorem 4 in Luz (1995). ■

In Section 4 we will use this characterization to show that random graphs with planted cliques are not Q graphs.

Inspired by the computational simplicity of the **LS** labelling, we study it more closely. As a labelling is completely defined by the associated kernel matrix, we refer to the following matrix

$$\mathbf{K} = \frac{\mathbf{A}}{\rho} + \mathbf{I} \text{ where } \rho \geq -\lambda_n(\mathbf{A}) \tag{2}$$

as the **LS** labelling. The KKT conditions for $\omega(\mathbf{K})$ in (1) are given by

$$\alpha_i + \frac{1}{\rho} \sum_{(i,j) \in E} A_{ij}\alpha_j = 1 + \mu_i,\ \mu_i\alpha_i = 0,\ \mu_i \geq 0. \tag{3}$$

Direct algebra yields

$$\omega(\mathbf{K}) = \sum_{i=1}^{n} \alpha_i^* \tag{4}$$

when α^* satisfies the KKT conditions.

2.2 On Regular Graphs

Before ending this section we would like to discuss the case of regular graphs. A graph is said to be $d-$regular if all nodes haves the same degree d. For such graphs one can compute an upper-bound, popularly known as Hoffman bound, on $\mathbf{ALPHA}(G)$ via the minimum eigenvalue of G. More specifically,

Theorem 6 (Hoffman and Howes, 1970) *For a d-regular graph, G, on n vertices and adjacency matrix $\mathbf{A}$,*

$$\mathbf{ALPHA}(G) \leq \frac{n}{1 - \frac{d}{\lambda_n}}$$

where λ_n is the smallest eigenvalue of $\mathbf{A}$.

Computation of this bound involves solving an eigenvalue problem, a significantly cheaper option than computing the ϑ function. However ϑ yields a tighter bound than the Hoffman bound. It was indeed proved (Lovász, 1979) that

$$\vartheta(G) \leq \frac{n}{1 - \frac{d}{\lambda_n}}.$$

It is interesting to note that $\omega(\mathbf{K})$ equals the Hoffman bound on d-regular graphs. To this end we have,

Lemma 7 *Let $G = (V,E)$, be a $d-$regular graph with adjacency matrix* $\mathbf{A}$*, and* $|V| = n$*. Then,*

$$\omega(\mathbf{K}) = \frac{n}{1+\frac{d}{\rho}}$$

where $\mathbf{K}$ *and* ρ *are defined by* (2)*.*

This was also derived in Luz (1995) as Corollary 1. To make the paper self-contained we give a short proof.

Proof For any d-regular graph the largest eigenvalue is d and the corresponding eigenvector is $\mathbf{e}$. It is easy to verify that $\alpha_i = \frac{1}{1+\frac{d}{\rho}}$ satisfies KKT conditions (3) and hence by (4) the lemma is proved. ■

The lemma holds for any $\rho \geq -\lambda_n$ and consequently, for regular graphs, $\omega(\mathbf{K})$ equals the Hoffman bound whenever $\rho = -\lambda_n$.

Before we discuss the applicability of the results obtained here to random graphs we study the interesting problem of finding a dense common subgraph in multiple graphs.

3. Finding Large Dense Regions in Multiple Graphs

The problem of finding a dense subgraph in a single graph is a computationally challenging problem (Pardalos and Rebennack, 2010). In this section we attempt an even more difficult version of the problem, namely that of finding a *common* large dense region in multiple graphs. Most methods which apply to single graphs (Lee et al., 2010) do not extend to the case of multiple graphs. An interesting method was proposed by Jiang and Pei (2009), which uses an enumerative strategy for finding a common dense subgraph in multiple graphs. This section presents one of the main contributions of this paper. It introduces the notion of *common orthogonal labelling* and proposes a Multiple Kernel Learning (MKL) (Lanckriet et al., 2004) inspired formulation. Later on we will see experimentally how this formulation achieves an order of magnitude scalability when compared with Jiang and Pei (2009).

3.1 Dense Common Subgraph Detection

Jiang and Pei (2009) studied the problem of finding *all* possible common subgraphs in graphs $G^{(1)},\ldots,G^{(m)}$ for a given choice of parameters $\gamma^{(1)},\ldots,\gamma^{(m)}$; such that the subgraphs have density at least $\gamma^{(l)}$ in graph $G^{(l)}$ for all $l \in \{1,\ldots,m\}$ respectively. The maximal quasiclique size depends on the choice of parameters $\gamma^{(1)},\ldots,\gamma^{(m)}$. In fact, if the parameters are not chosen properly the algorithm may fail to return any solution at all. A different choice of parameters requires solving the problem again. As a consequence one might have to run several iterations of the algorithm with different parameter choices before obtaining a desired subgraph.

The approach is essentially enumerative in nature, and consequently, the space and time complexity of the algorithm is exponential in the size of the largest possible clique, rendering it impractical for finding large dense regions. For example, finding subgraphs of size 60 requires 11.5 hours (see Figure 17 in Jiang and Pei, 2009). Clearly this algorithm is not suitable for finding large subgraphs.

In practice one might wish to quickly find a single subset of vertices which is large (some fraction of the overall graph) and which induces a dense subgraph in each of the original graphs

$G^{(l)} \in \mathbb{G}$ without fine tuning of parameters; or multiple runs of the algorithm. In other words, one would like to have a subgraph finding algorithm which is *parameter-less*. To this extent, we define the problem formally as follows:

Definition 8 (Problem definition) *Let* $\mathbb{G} := \{G^{(1)}, \ldots, G^{(m)}\}$ *be a set of simple, undirected graphs* $G^{(l)} = (V, E^{(l)})$ *defined on vertex set* $V = \{1, \ldots, n\}$. *Find a* common *subgraph which is* dense *in all the graphs.*

The remainder of this section is organized as follows: we develop the notion of *common orthogonal labelling* and establish the connection to MKL formulation in Section 3.1.1. Section 3.1.2 presents our algorithm for recovery of large common dense subgraph from multiple graphs.

3.1.1 COMMON ORTHOGONAL LABELLING AND MKL FORMULATION

We begin by defining common orthogonal labelling below:

Definition 9 *Given set of simple undirected graphs* $\mathbb{G}$ *on a common vertex set* $V = \{1, \ldots, n\}$, *the common orthogonal labelling* $\mathbf{u}_1, \ldots, \mathbf{u}_n$ *is given by* $\mathbf{u}_i \in \mathcal{S}^{d-1}$ *such that* $\mathbf{u}_i^\top \mathbf{u}_j = 0$ *if* $(i, j) \notin E^{(l)}\ \forall l = \{1, \ldots, m\}$.

Let $\mathcal{K}(\mathbb{G})$ denote the set of common orthogonal labellings of $\mathbb{G}$ so that

$$\mathcal{K}(\mathbb{G}) = \left\{\mathbf{K} : \mathbf{K} \in \mathbf{S}_+^n,\ \mathbf{K}_{ij} = 0\ if\ (i, j) \notin E^{(l)}\ \forall 1 \leq l \leq m\right\}.$$

The common orthogonal labellings of $\mathbb{G}$ is related to the *union graph* $G^\cup = (V, E^\cup)$ constructed from $\mathbb{G}$ as follows: an edge (i, j) is present in the union graph $G^\cup$ if it is present in at least one of the original graphs $G^{(l)} \in \mathbb{G}$, and absent otherwise, that is, $(i, j) \notin E^\cup$ iff $(i, j) \notin E^{(l)} \forall_l$. By construction, we see

$$\mathcal{K}(\mathbb{G}) = \mathcal{K}(G^\cup).$$

Let $\Upsilon(\mathbb{G})$ denote the size of the *maximum common independent set*, that is, subset of vertices $CS \subseteq V$ of maximum possible cardinality for which the subgraph $G_{CS}^{(l)}$ induced by CS in graph $G^{(l)}$ is an independent set for all $G^{(l)} \in \mathbb{G}$. It is immediate that $\Upsilon(\mathbb{G})$ is equal to the size of the maximum independent set in the union graph $\mathbf{ALPHA}(G^\cup)$. Following the arguments in Theorem 3, one can show

$$\mathbf{ALPHA}(G^\cup) = \Upsilon(\mathbb{G}) \leq \vartheta(G^\cup) \leq \omega(\mathbf{K})\ \forall\, \mathbf{K} \in \mathcal{K}(\mathbb{G}). \tag{5}$$

As noted in the previous section, the optimization problem in (5) is a SDP, which cannot be solved for large graphs. Let $\mathbf{K}^{(l)} = \frac{\mathbf{A}^{(l)}}{\rho^{(l)}} + \mathbf{I}$ and $\rho^{(l)} \geq -\lambda_n(\mathbf{A}^{(l)})$ be the **LS** labelling for graph $G^{(l)}$. In the remainder of this section, we use the notation $\mathbb{K} := \{\mathbf{K}^{(1)}, \ldots, \mathbf{K}^{(m)}\}$ to denote the set of orthogonal labellings corresponding to the graphs $\mathbb{G}$.

We consider the set of convex combinations of the labellings of the original graphs $\mathbb{K}$ such that

$$conv(\mathbb{K}) := \left\{\mathbf{K} : \mathbf{K} = \sum_{l=1}^{m} \delta^{(l)} \mathbf{K}^{(l)} \text{ with } \delta^{(l)} \geq 0,\ \sum_{l=1}^{m} \delta^{(l)} = 1\right\}.$$

Note that $K_{ij}^{(l)}$ is zero whenever edge (i, j) is absent in $G^\cup$ for all graphs $G^{(l)}$ and consequently $\mathbf{K}_{ij} = 0$. Thus $\mathbf{K} \in conv(\mathbb{K})$ is a common orthogonal labelling and the following is immediate

$$conv(\mathbb{K}) \subseteq \mathcal{K}(\mathbb{G}).$$

Instead of solving the original problem, we consider the following problem

$$\psi(\mathbb{K}) = \min_{\mathbf{K} \in conv(\mathbb{K})} \omega(\mathbf{K}). \tag{6}$$

An immediate advantage of this formulation over the SDP formulation is that it could be solved efficiently by standard MKL solvers (Rakotomamonjy et al., 2008; Aflalo et al., 2011). The quantity, $\psi(\mathbb{K})$ also defines an upper bound on the size of the maximum common independent set. More precisely

$$\Upsilon(\mathbb{G}) \leq \min_{\mathbf{K} \in \mathcal{K}(\mathbb{G})} \omega(\mathbf{K}) \leq \psi(\mathbb{K}).$$

Rewriting (6), we get

$$\begin{aligned}
\psi(\mathbb{K}) &= \min_{\mathbf{K} \in conv(\mathbb{K})} \omega(\mathbf{K}) = \min_{\delta^{(l)} \geq 0, \sum \delta^{(l)} = 1} \omega\big(\sum_l \delta^{(l)} \mathbf{K}^{(l)}\big) \\
&= \min_{\delta^{(l)} \geq 0, \sum \delta^{(l)} = 1} \max_{\alpha \geq 0} \sum_l \delta^{(l)} f(\alpha; \mathbf{K}^{(l)}) \\
&= \max_{\alpha \geq 0} \min_{\delta^{(l)} \geq 0, \sum \delta^{(l)} = 1} \sum_l \delta^{(l)} f(\alpha; \mathbf{K}^{(l)}).
\end{aligned}$$

The optimization problem is linear in δ and strictly feasible in both δ and α. One can interchange $\min_\delta \max_\alpha$ to $\max_\alpha \min_\delta$, Sion (1958) yielding the last equality. For any vector, $\mathbf{x} = [x_1, \ldots, x_d]^\top$,

$$\min_{0 \leq \delta_i \leq 1, \sum_{i=1}^d \delta_i = 1} \delta^\top x = \min(x_1, \ldots, x_d) = \max_{t, x_i \geq t} t.$$

An alternative re-statement of (6) is

$$\psi(\mathbb{K}) = \max_{t \in \mathbb{R}, \alpha_i \geq 0} t \quad \text{s.t.} \;\; f(\alpha; \mathbf{K}^{(l)}) \geq t \; \forall \; 1 \leq l \leq m. \tag{7}$$

The Lagrange dual of (7) is given by

$$\mathcal{L}(t, \alpha, \lambda, \delta) = t + \sum_{l=1}^m \lambda_l \big(f(\alpha; \mathbf{K}^{(l)}) - t\big) + \sum_{i=1}^n \delta_i \alpha_i,$$

where $\lambda \in \mathbb{R}_+^m$ and $\delta \in \mathbb{R}_+^n$ denote the dual variables. The KKT conditions yield

$$2 \sum_{l=1}^m \lambda_l^* \big(1 - \alpha_i^* - \frac{1}{\rho^{(l)}} \sum_j A_{ij}^{(l)} \alpha_j^*\big) + \delta_i^* = 0, \quad \delta_i^* \alpha_i^* = 0, \quad \delta_i^*, \alpha_i^* \geq 0 \forall i \in [n], \quad \text{and} \tag{8}$$

$$\lambda_l^* (f(\alpha^*; \mathbf{K}^{(l)}) - t^*) = 0, \quad \sum_l \lambda_l^* = 1, \quad \lambda_l^* \geq 0 \forall l \in [m]. \tag{9}$$

The above optimization can be readily solved by state of the art MKL solvers. The obvious question that arises is when, or precisely, for which sets of graphs does solving (6) yield good approximation to the original problem of computing $\Upsilon(\mathbb{G})$.

In order to address this, we begin by defining the family of *Common Quadratically-stable* ($\mathcal{CQ}$) sets of graphs.

Definition 10 *A set of graphs $\mathbb{G} = \{G^{(1)}, \ldots, G^{(m)} : G^{(i)} = (V, E^{(i)})\}$ having a common vertex set V is Common Quadratically-stable if the optimal value $\psi(\mathbb{K})$ in* (6) *is equal to the size of the maximum common independent set $\Upsilon(\mathbb{G})$, that is,*

$$\Upsilon(\mathbb{G}) = \psi(\mathbb{K}), \tag{10}$$

where $\mathbb{K}$ is the set of **LS** *labellings of graphs $\mathbb{G}$.*

Remark 11 *We use the notation $\mathbb{G} \in \mathcal{CQ}$ to denote that the set of graphs $\mathbb{G}$ satisfies* (10)*, that is, it is Common Quadratically-stable.*

Let $Y \subseteq V$ denote the maximum common independent set in $\mathbb{G}$; with indicator vector $\mathbf{e}_Y$. We now characterize $\mathcal{CQ}$ family (of sets of graphs having a common vertex set) in the following result.

Theorem 12 *Given set of graphs $\mathbb{G} = \{G^{(1)}, \ldots, G^{(m)}\}$ having common vertex set V and* **LS** *labellings $\mathbf{K}^{(l)} = \frac{\mathbf{A}^{(l)}}{\rho^{(l)}} + \mathbf{I}$ and $\rho^{(l)} \geq -\lambda_n(\mathbf{A}^{(l)})\ \forall l \in [m]$; the optimal value $\psi(\mathbb{K})$ in* (6) *is equal to the size of the maximum common independent set Y, that is, the set $\mathbb{G}$ is Common Quadratically-stable ($\mathbb{G} \in \mathcal{CQ}$) if there exists non-empty $\mathbb{G}_Q \subseteq \mathbb{G}$ such that*

$$\rho^{(l)} \leq \min_{i \notin CS} |N_i(G^{(l)}) \cap Y| \qquad \forall G^{(l)} \in \mathbb{G}_Q. \tag{11}$$

Proof Let $L := |\mathbb{G}_Q|$ denote the number of graphs which satisfy the property

$$\rho^{(l)} \leq \min_{i \notin Y} |N_i(G^{(l)}) \cap Y| \quad \forall G^{(l)} \in \mathbb{G}_Q.$$

We consider primal solution $\alpha^* = \mathbf{e}_Y$ and dual solution given by

$$\lambda_l^* = \begin{cases} \frac{1}{L} & \text{if } l : G^{(l)} \in \mathbb{G}_Q \\ 0 & \text{otherwise} \end{cases}, \text{ and } \quad \delta_i^* = \begin{cases} 0 & \text{if } i \in Y \\ \sum_l 2\lambda_l^* (1 - \frac{|N_i(G^{(l)}) \cap Y|}{\rho^{(l)}}) & \text{otherwise} \end{cases}.$$

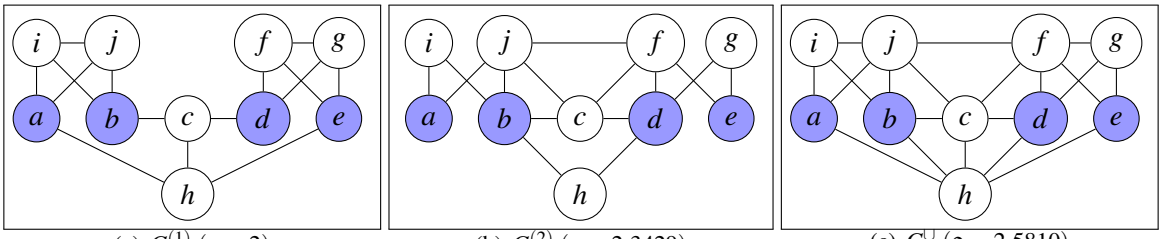
This solution satisfies the KKT conditions in (8) and (9). Further, the optimization in (7) is convex; and the KKT conditions are sufficient for optimality. The optimal value is given by $t^* = \sum_{i=1}^n \alpha_i^*$ where $\alpha^* = \mathbf{e}_Y$ yielding $\psi(\mathbb{K}) = \Upsilon(\mathbb{G})$. ■

Recall as in (5) that $\min_{\mathbf{K} \in \mathcal{K}(\mathbb{G})} \omega(\mathbf{K})$ is equivalent to computing Lovász ϑ function of $G^\cup$. Therefore, one can easily solve it whenever $G^\cup$ is a $\mathcal{Q}$ graph by solving $\omega(\mathbf{K}^\cup)$ where $\mathbf{K}^\cup = \frac{A^\cup}{\rho} + \mathbf{I}$ and $\rho \geq -\lambda_n(\mathbf{A}^\cup)$ whenever $\rho \leq \min_{i \notin Y} |N_i(G^\cup) \cap Y|$. It is of interest to find when $G^\cup$ is *not* a $\mathcal{Q}$ graph yet $\mathbb{G}$ is Complex Quadratically-stable ($\mathbb{G} \in \mathcal{CQ}$). One can easily see by construction of $G^\cup$ that

$$\min_{i \notin Y} |N_i(G^{(l)}) \cap Y| \leq \min_{i \notin Y} |N_i(G^\cup) \cap Y| \quad \forall G^{(l)} \in \mathbb{G}.$$

However, the relationship between minimum eigenvalue $\rho^{(l)} = -\lambda_n(\mathbf{A}^{(l)})$ of original graphs $G^{(l)} \in \mathbb{G}$, and minimum eigenvalue $\rho^\cup = -\lambda_n(A^\cup)$ of union graph $G^\cup$ is not clear, that is, $\rho^\cup$ can be greater or less than $\rho^{(l)}$ (see, e.g., Brouwer and Haemers, 2012, 3.1-3.2). We illustrate this in the following example.

Example 1 *Consider the graphs $G^{(1)}$ and $G^{(2)}$ shown in Figure 1. The maximum common independent set is $Y = \{a, b, d, e\}$ and $\min_{i \notin Y} |N_i \cap Y| = 2$. One can show that $\mathbb{G} \in \mathcal{CQ}$ since $\rho^{(1)} \leq \min_{i \notin Y} |N_i \cap Y|$, while the union graph $G^\cup$ is not a $\mathcal{Q}$ graph. Therefore, one can compute $\Upsilon(\mathbb{G})$ by solving the MKL optimization in* (7) *even though the maximum independent set problem in $G^\cup$ cannot be solved using QP. Thus, the MKL formulation is advantageous whenever that there exists some graph $G^{(l)} \in \mathbb{G}$ which satisfies* (11)*, even though $G^\cup$ is not a $\mathcal{Q}$ graph.*



(a) $G^{(1)}$ ($\rho = 2$) (b) $G^{(2)}$ ($\rho = 2.3429$) (c) $G^{\cup}$ ($\rho = 2.5810$)

Graph(s)	ρ	$\min_{i \notin Y} \lvert N_i \cap Y \rvert$	$\omega(\mathbf{K})$	$\psi(\mathbb{K})$
$G^{\cup}$	2.5810	2	4.3189	-
$\mathbb{G}$	-	-	-	**4.0**

Figure 1: Example of $\mathcal{CQ}$ set of graphs. Here $\mathbb{G} = \{G^{(1)}, G^{(2)}\}$ and maximum common independent set is $Y = \{a,b,d,e\}$ (highlighted in blue). The graph $G^{(1)}$ satisfies $\rho \leq \min_{i \notin Y} |N_i \cap Y|$ in (11); while $G^{(2)}$ and $G^{\cup}$ do not. Hence $\mathbb{G} \in \mathcal{CQ}$, while union graph $G^{\cup}$ is not a $\mathcal{Q}$ graph. The MKL optimization yields $\psi(\mathbb{K}) = 4$ which is equal to size of maximum common independent set $Y = \{a,b,d,e\}$.

3.1.2 SUBGRAPH DETECTION BASED ON MULTIPLE GRAPHS

In the remainder of this section, we relate the optimal solution (support vectors) of $\omega(\mathbf{K})$ and the density of related induced subgraph for the single graph case; which we later extend to multiple graphs. We first recall an interesting property of optimal solution α^* which maximizes $f(\alpha;\mathbf{K})$ in (1) when G is a $\mathcal{Q}$ graph.

Remark 13 (Luz, 1995) *Let $G = (V,E)$ be a $\mathcal{Q}$ graph having unique maximum independent set $S \subseteq V$. Then, the optimal solution α^* which maximizes objective $f(\alpha;\mathbf{K})$ in (1) is given by $\alpha^* = \mathbf{e}_S$, that is,*

$$\alpha_i^* = \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{otherwise} \end{cases}.$$

The above claim follows from observing that α^* satisfies the Karush-Kuhn-Tucker (KKT) conditions, which is sufficient for optimality. The key idea here is that choosing the vertices having support vectors with high numerical values yields a subgraph with low density (or more precisely, independent set with zero density for $\mathcal{Q}$ graphs). We extend this notion to general graphs by relating the density of the induced subgraph obtained by choosing vertices having "high" support through the KKT conditions.

We now consider a general graph $G = (V,E)$ with adjacency matrix $\mathbf{A}$, and let α^* be the optimal solution of (1) when $\mathbf{K} = \frac{\mathbf{A}}{\rho} + \mathbf{I}$ with $\rho \geq -\lambda_n(\mathbf{A})$. We wish to relate the density of the subgraph G_{S_c} induced by the "high" support vectors $S_c := \{i : \alpha_i^* \geq c\}$ for some threshold $c \in (0,1)$.

Let $\bar{\alpha}_i^*(S)$ denote the average of the support vectors α_j^* over the neighbourhood $N_i(G_S)$ of node i in subgraph G_S induced by $S \subseteq V$ in graph G; and $\bar{\alpha}_S^*$ be the minimum $\bar{\alpha}_i^*(S)$ over all $i \in S$, that is,

$$\bar{\alpha}_i^*(S) = \frac{\sum_{j \in S} A_{ij} \alpha_i^*}{d_i(G_S)}, \quad \text{and} \quad \bar{\alpha}_S^* = \min_{i \in S} \bar{\alpha}_i^*(S). \tag{12}$$

With the above notation, one can show the following:

Lemma 14 *Let $G = (V,E)$ be a simple graph with at least one edge. Let α^* be the optimal solution of* (1) *where $\mathbf{K} = (\frac{\mathbf{A}}{\rho} + \mathbf{I})$ denotes the* **LS** *labelling of G and $\rho \geq -\lambda_n(\mathbf{A})$. The set $S_c = \{i : \alpha_i^* > c\}$ with cardinality n_c induces a subgraph G_{S_c} with density $\gamma(G_{S_c})$ where*

$$\gamma(G_{S_c}) \leq \frac{\rho(1-c)}{\bar{\alpha}^*_{S_c}(n_c-1)} \leq \frac{\rho(1-c)}{c(n_c-1)}.$$

Proof The Karush-Kuhn Tucker (KKT) conditions for (1) are given in (3). Using definition in (12), one can obtain

$$\bar{\alpha}^*_{S_c} \sum_{i \in S_c} d_i(G_{S_c}) \leq \sum_{i \in S_c} \bar{\alpha}^*_i(S_c) d_i(G_{S_c}) = \sum_{i,j \in S_c} \mathbf{A}_{ij} \alpha^*_j \leq \rho(n_c - \sum_{i \in S_c} \alpha^*_i) \leq \rho n_c(1-c).$$

Observing that $\sum_{i \in S_c} d_i(G_{S_c}) = 2|E_{S_c}|$, and dividing by $\binom{n_c}{2}$ yields the desired result, that is,

$$\gamma(G_{S_c}) \leq \frac{\rho(1-c)}{\bar{\alpha}^*_{S_c}(n_c-1)} \leq \frac{\rho(1-c)}{c(n_c-1)} \quad (\because \bar{\alpha}^*_{S_c} \geq c \text{ by defn. in (12)}).$$

■

This result provides an upper bound on the density $\gamma(G_{S_c})$ of the subgraph induced by set S_c in graph G for general c. Two special cases of interest are the set of non-zero support vectors $SV = \{i : \alpha_i^* > 0\}$ and the set of support vectors with support one $S_1 := \{i : \alpha_i^* = 1\}$ respectively.

Setting $c = 0$, one can show the set of (non-zero) support vectors $SV = \{i : \alpha_i^* > 0\}$ with cardinality $n_{SV} = |SV|$ induces a subgraph G_{SV} having density $\gamma(G_{SV})$ at most $\rho/\bar{\alpha}^*_{SV}(n_{SV}-1)$, that is,

$$\gamma(G_{SV}) \leq \frac{\rho}{\bar{\alpha}^*_{SV}(n_{SV}-1)}.$$

This provides a simple procedure for finding a sparse subgraph by selection the subgraph G_{SV} induced by the set of non-zero support vectors SV. It also gives an upper bound on density.

Setting c arbitrarily close to 1, one can show the set $S_1 = \{i : \alpha_i^* = 1\}$ of support vectors having support 1 is an independent set in G. An extreme case is when G is a Q graph and all support vectors are 1 over the maximum independent set and 0 otherwise.

We now consider the problem of common dense subgraph recovery from multiple graphs based on the MKL formulation in (7). Let (α^*, t^*) be the optimal solution of (7). One can partition the set $\mathbb{G}$ into two sets: *active* graphs $\mathbb{G}_A \subseteq \mathbb{G}$ for which the constraint in (7) is tight, that is,

$$f(\alpha^*, \mathbf{K}^{(l)}) = t^* \quad \forall\, G^{(l)} \in \mathbb{G}_a,$$

and *inactive* graphs $\mathbb{G}_i := \mathbb{G} \backslash \mathbb{G}_a$, where the constraint is not tight. Consequently, the analysis in the single graph case based on selection of support vector above certain threshold cannot be directly extended to multiple graph case. In the remainder, we address this using a two step procedure.

Let $SV := \{i : \alpha_i^* > 0\}$ and $S_c := \{i : \alpha_i^* > c\}$ denote the set of support vectors, and the set of support vectors having "high" support respectively for some appropriate choice of $c \in [0,1)$. Let

$\bar{\alpha}_i^{(l)}(S)$ denote the average of the support vectors α_j^* over the neighbourhood $N_i(G_S^{(l)})$ of node i in subgraph $G_S^{(l)}$ induced by $S \subseteq V$ in graph $G^{(l)}$; and $\bar{\alpha}_S^*$ be the minimum $\bar{\alpha}_i^*(S)$ over all $i \in S$, that is,

$$\bar{\alpha}_i^{(l)}(S) = \frac{\sum_{j\in S} A_{ij}\alpha_i^*}{d_i(G_S^{(l)})}, \quad \text{and,} \quad \bar{\alpha}_S^{(l)} = \min_{i\in S} \bar{\alpha}_i^{(l)}(S).$$

Notice that since $\bar{\alpha}_S^{(l)}$ is minimum (over all vertices in the set S) of average value of support vectors in the neighbourhood $N_i(S)$ for a vertex $i \in S$, it is equal or greater than the minimum support vector $\min_{i\in S}\alpha_i^*$ in S. Let $\alpha_{\min} = \min_{i\in SV}\alpha_i^*$ denote the minimum non-zero support vector. We define the sets $T^{(l)} \subseteq V$ and $T \subseteq V$ as

$$T^{(l)} := T^{(l)}(SV) = \left\{ i \in SV : 1 - \alpha_l - d_i(G_{SV}^{(l)})\bar{\alpha}_{SV}^{(l)}/\rho^{(l)} > 0 \right\}, \quad T = \cap_{l=1}^m T^{(l)}. \tag{13}$$

Then, one can show the following

Lemma 15 *Given set of graphs $\mathbb{G} = \{G^{(l)} : G^{(l)} = (V, E^{(l)}) \forall\, l \in [m]\}$ defined on common vertex set V with* **LS** *labelling $\mathbb{K}$ where $\mathbf{K}^{(l)} = \frac{\mathbf{A}^{(l)}}{\rho^{(l)}} + \mathbf{I}$ and $\rho^{(l)} \geq -\lambda_n(\mathbf{A}^{(l)})$. Let (α^*, t^*) be the optimal solution of* (7)*, and T be defined as in* (13)*. The set T with cardinality $n_T = |T|$ induces a subgraph $G_T^{(l)}$ in graph $G^{(l)} \in \mathbb{G}$ having density at most $\gamma(G_T^{(l)})$ given by*

$$\gamma(G_T^{(l)}) \leq \frac{(1-\alpha_{\min})\rho^{(l)}}{\bar{\alpha}_{SV}^{(l)}(n_T - 1)} \quad \forall\, G^{(l)} \in \mathbb{G}.$$

Proof The KKT conditions in (8) and (9) yield $\lambda_l^* = 0$ for $G^{(l)} \in \mathbb{G}_i$. Further,

$$\begin{aligned}
0 &= \sum_{i=1}^n \alpha_i^* \left(2\sum_{l=1}^m \lambda_l^* \big(1 - \alpha_i^* - \frac{1}{\rho^{(l)}} \sum_j A_{ij}^{(l)} \alpha_j^*\big) + \delta_i^* \right) \\
&= \sum_{l: G^{(l)} \in \mathbb{G}_a} \lambda_l^* \sum_i \alpha_i^* \big(1 - \alpha_i^* - \frac{1}{\rho^{(l)}} \sum_j A_{ij}^{(l)} \alpha_j^*\big) && (\because \alpha_i^* \delta_i^* = 0 \forall_i) \\
&= \sum_{l: G^{(l)} \in \mathbb{G}_a} \lambda_l^* (t^* - \sum_i \alpha_i^*) = t^* - \sum_i \alpha_i^* && (\because \sum_l \lambda_l^* = 1).
\end{aligned}$$

This yields $t^* = \sum_i \alpha_i^*$, that is, the objective is equal to sum of the support vectors. We can rewrite feasibility condition $f(\alpha^*; \mathbf{K}^{(l)}) \geq t^*$ for all graphs $G^{(l)} \in \mathbb{G}$:

$$\begin{aligned}
0 &\leq \sum_{i\in SV} \alpha_i^* (2 - \alpha_i^* - \sum_{j\neq i} A_{ij}^{(l)} \alpha_j^*) - t^* \quad = \sum_{i\in SV} \alpha_i^* (1 - \alpha_i^* - \sum_{j\neq i} A_{ij}^{(l)} \alpha_j^*) \\
&= \sum_{i\in T} \alpha_i^* \underbrace{\big(1 - \alpha_i^* - \frac{d_i(G_{SV}^{(l)})}{\rho^{(l)}} \bar{\alpha}_{SV}^{(l)}\big)}_{>0} + \sum_{i\notin T} \alpha_i^* \underbrace{\big(1 - \alpha_i^* - \frac{d_i(G_{SV}^{(l)})}{\rho^{(l)}} \bar{\alpha}_{SV}^{(l)}\big)}_{<0} \\
&\leq \sum_{i\in T} \alpha_i^* \big(1 - \alpha_i^* - \frac{d_i(G_{SV}^{(l)})}{\rho^{(l)}} \bar{\alpha}_{SV}^{(l)}\big) \quad \leq \sum_{i\in T} \alpha_i^* \big(1 - \alpha_i^* - \frac{d_i(G_T^{(l)})}{\rho^{(l)}} \bar{\alpha}_{SV}^{(l)}\big) \\
&\leq \sum_{i\in T} \alpha_i^* \big(1 - \alpha_{\min} - \frac{d_i(G_T^{(l)})}{\rho^{(l)}} \bar{\alpha}_{SV}^{(l)}\big) \quad \leq \sum_{i\in T} \big(1 - \alpha_{\min} - \frac{d_i(G_T^{(l)})}{\rho^{(l)}} \bar{\alpha}_{SV}^{(l)}\big) \\
&\leq n_T(1 - \alpha_{\min}) - \frac{\bar{\alpha}_{SV}^{(l)}}{\rho^{(l)}} \sum_{i\in T} d_i(G_T^{(l)}) \quad = n_T(1 - \alpha_{\min}) - \frac{\bar{\alpha}_{SV}^{(l)}}{\rho^{(l)}} \gamma(G_T^{(l)}) n_T(n_T - 1).
\end{aligned}$$

Algorithm 1 $T = CSS(G^{(1)}, \ldots, G^{(M)})$

Get α^* using MKL solver to solve eqn. (7)
$T = \cap_{l=1}^{m} T^{(l)}$ {eqn. (13)}
Return T

Dividing by $\binom{n_T}{2}$ and rewriting, we get

$$\gamma(G_T^{(l)}) \leq \frac{(1-\alpha_{\min})\rho^{(l)}}{\bar{\alpha}_{SV}^{(l)}(n_T-1)},$$

which completes the proof. ■

The above result allows us to build a *parameter-less* common sparse subgraph (CSS) algorithm shown in Algorithm 1 having following advantages: it provides a theoretical bound on subgraph density; and, it requires no parameters from the user beyond the set of graphs $\mathbb{G}$.

The size of the induced subgraph n_T is important in the overall quality of the solution. Ideally, one would like n_T to be some large fraction of the overall number of nodes N, typically $n_T/N \leq 1/2$. However, if n_T is very large, that is, $n_T/N \simeq 1$, the density of the induced subgraph is close to the average graph density. More generally, one might be interested in a trade-off between the subgraph size n_T and subgraph density $\gamma(G_T^{(l)})$. Analogous to the simple graph case, we can improve the subgraph density is obtained by choosing smaller region nodes $T_c := \{i \in T : \alpha_i^* > c\} \subseteq T$. We discuss this further in Section 5.2.

4. SVM $-$ ϑ Graphs: Graphs Where $\vartheta(G)$ Can Be Approximated by SVM

Computing the Lovász function and related relaxations involves solving a semidefinite program. Off the shelf SDP solvers are computationally very demanding and do not scale to graphs of more than 5000 vertices. In this section we study Erdös-Rényi graphs, parametrized as $G(n,p)$ where p is probability of an edge and n is the number of vertices. As noted in Section 2, one can evaluate the ϑ function in $O(n^3)$ on $\mathcal{Q}$ graphs. Further, we have equality $\mathbf{ALPHA}(G) = \vartheta(G) = \omega(\mathbf{K})$ for $\mathcal{Q}$ graphs, where $\mathbf{K}$ is a **LS** labelling. It is well known that in $G(n,1/2)$ graph, with high probability, $\mathbf{ALPHA}(G) = \Theta(\log n)$ whereas $\vartheta(G) = \Theta(\sqrt{n})$ (Juhász, 1982; Coja-Oghlan, 2005). This immediately establishes a negative result that $G(n,1/2)$ graphs are not $\mathcal{Q}$ graphs. In the following we will show that despite the above negative result one can still obtain constant factor approximation to ϑ function on random graphs by solving a SVM problem. We begin by introducing a class of graphs, called $\mathbf{SVM}-\vartheta$ graphs, where the gap between ϑ and $\omega(\mathbf{K})$ for $\mathbf{K}$ defined by (2) is not too large. Subsequently we show that $G(n,p)$ graphs and $G(n,p)$ graphs with planted cliques are $\mathbf{SVM}-\vartheta$ graphs. This results immediately show that one can identify planted cliques or planted subgraphs. Furthermore we prove a concentration result on $\omega(\mathbf{K})$ for $G(n,p)$ graphs.

Definition 16 *A family of graphs $\mathcal{G} = \{G = (V,E)\}$ is said to be $\mathbf{SVM}-\vartheta$ graph family if there exist a constant γ, such that for any graph $G \in \mathcal{G}$ with $|V| \geq n_0$, the following holds:*

$$\omega(\mathbf{K}) \leq \gamma\vartheta(G),$$

where $\omega(\mathbf{K})$ is defined in (1) *and $\mathbf{K}$ is defined on G by* (2)*.*

Such classes of graphs are interesting because of two reasons. Firstly on these class of graphs one can approximate the Lovász function well without resorting to solving a SDP, and secondly the ϑ function in turn can be used in the design and analysis of approximation algorithms. We will demonstrate examples of such families of random graphs: the Erdös–Rényi random graph $G(n,p)$ and a planted variation. Here the relaxation $\omega(\mathbf{K})$ could be used in place of $\vartheta(G)$, resulting in algorithms with the same quality guarantees but with faster running time—in particular, this will allow the algorithms to be scaled to large graphs.

4.1 $G(n,p)$ Graphs Are SVM $-\vartheta$ Graphs

In this section we show that $G(n,p)$ graphs are indeed **SVM** $-\vartheta$. We begin with some preliminaries.

4.1.1 PRELIMINARIES

The following lemma is well known (see Boyd and Vandenberghe, 2004, Section 9.1.2):

Lemma 17 *A function $g : C \subset \mathbb{R}^d \to \mathbb{R}$ is said to be strongly concave over C if there exists $t > 0$ such that $\nabla^2 g(\mathbf{x}) \preceq -t\mathbf{I}\ \forall\ \mathbf{x} \in C$. For such functions one can show that if $p^* = g(\mathbf{x}^*) = \max_{\mathbf{x}\in C} g(\mathbf{x}) < \infty$ then*

$$\forall \mathbf{x} \in C: \quad \frac{t}{2}\|\mathbf{x}-\mathbf{x}^*\|^2 \ \le\ p^* - g(\mathbf{x}) \ \le\ \frac{1}{2t}\|\nabla g(\mathbf{x})\|^2.$$

The classical Erdös-Rényi random graph $G(n,p)$ has n vertices and each edge (i,j) is present independently with probability p. (In the closely related the $G(n,M)$ model, a graph is chosen uniformly at random from the collection of all graphs which have n nodes and M edges, or, equivalently, a set of M edges is chosen uniformly at random without replacement from the set of all possible $\binom{n}{2}$ edges. With $M = \binom{n}{2}p$, the two models are essentially equivalent.) For many types of random distributions, $G(n,p)$ is considered a paradigm choice for input instances and hence both the combinatorial structure and the algorithmic theory of $G(n,p)$ are of fundamental interest (Bollobás, 2001; Janson et al., 2000; Frieze and McDiarmid, 1997). We list a few well known facts about $G(n,p)$ that will be used repeatedly.

Remark 18 *For $G(n,p)$ for any $0 \le p < 1$,*

- *With probability $1 - O(1/n)$, the degree of each vertex is in the range $np \pm \sqrt{np\log n}$.*
- *With probability $1 - e^{-n^c}$ for some $c > 0$, the maximum eigenvalue is $np(1+o(1))$. The the minimum eigenvalue is in the range $[-2\sqrt{np(1-p)}, 2\sqrt{np(1-p)}]$ (Füredi and Komlós, 1981),*
- *With high probability, $\vartheta(G(n,p)) = \Theta(\sqrt{\frac{n(1-p)}{p}})$ [Coja-Oghlan, 2005; Juhász, 1982].*

4.1.2 $G(n,p)$ GRAPHS ARE SVM $-\vartheta$

We are now ready to state our main result.

Theorem 19 *Let $G = G(n,p)$, with $p(1-p) = \Omega(n^{-1}\log^4 n)$. For every constant $\delta > 0$,*

$$\omega(\mathbf{K}) \le (1+O(1))\vartheta(G)$$

holds with probability $1 - O(1/n)$, whenever $\mathbf{K}$ is defined in (2) with $\rho = (1+\delta)2\sqrt{np(1-p)}$.

Proof By Remark 18 for all choices of $\delta > 0$, the minimum eigenvalue of $\frac{1}{\rho}\mathbf{A} + \mathbf{I}$ is, almost surely, greater than 0 which implies that $f(\alpha, \mathbf{K})$ (see (1)) is strongly concave. As $\mathbf{A}$ is random we begin by analyzing the KKT conditions (3) for $\mathbb{E}(\mathbf{A})$, the expectation of $\mathbf{A}$. For $G(n,p)$ graph $\mathbb{E}(\mathbf{A}) = p(\mathbf{e}\mathbf{e}^\top - \mathbf{I})$. For the given choice of ρ, the matrix $\tilde{\mathbf{K}} = \frac{\mathbb{E}(\mathbf{A})}{\rho} + \mathbf{I}$ is positive definite. More importantly $f(\alpha, \tilde{\mathbf{K}})$ is again strongly concave and attains maximum at a KKT point. By direct verification $\hat{\alpha} = \hat{\beta}\mathbf{e}$ where $\hat{\beta} = \frac{\rho}{(n-1)p+\rho}$ satisfies the KKT conditions. More precisely

$$\hat{\alpha} + \frac{1}{\rho}\mathbb{E}(\mathbf{A})\hat{\alpha} = \mathbf{e}. \tag{14}$$

Thus $\hat{\alpha}$ is the optimal for the expected case with the optimal value, $\bar{f}$, given by

$$\bar{f} = \max_{\alpha \geq 0} f(\alpha, \tilde{\mathbf{K}}) = 2\sum_{i=1}^{n} \hat{\alpha}_i - \hat{\alpha}^\top \left(\frac{\mathbb{E}(\mathbf{A})}{\rho} + \mathbf{I} \right) \hat{\alpha} = n\hat{\beta}. \tag{15}$$

By choice of ρ, for any p in the regime $np \geq 1$ one notes that

$$\hat{\beta} = \frac{\rho}{(n-1)p+\rho} = \frac{\rho}{np}(1 + o(1)) = 2(1+\delta)\sqrt{\frac{1-p}{np}}. \tag{16}$$

The last equality holds by neglecting the $o(1)$ term. Using the fact about degrees of vertices in $G(n,p)$, in the regime of interest,

$$\mathbf{a}_i^\top \mathbf{e} = (n-1)p + \Delta_i \text{ with } |\Delta_i| \leq \sqrt{np\log n}, \tag{17}$$

where $\mathbf{a}_i^\top$ is the i^{th} row of the adjacency matrix $\mathbf{A}$. It is interesting to note that $\hat{\alpha}$ is an approximate KKT point for (3). Indeed, for all $i \in V$, application of (17) and (16) alongwith the choice of ρ as given in the statement of theorem we obtain

$$\left| \hat{\alpha}_i + \frac{1}{\rho}\sum_j A_{ij}\hat{\alpha}_j - 1 \right| = \left| \frac{\hat{\beta}}{\rho}\Delta_i \right| \leq \sqrt{\frac{\log n}{np}}. \tag{18}$$

We would like to exploit this property to approximate the $\omega(\mathbf{K})$ for a random graph. To this end note that

$$f(\hat{\alpha}; \mathbf{K}) = \hat{\alpha}^\top \mathbf{e} + \sum_{i=1}^{n} \hat{\alpha}_i \left(1 - \hat{\alpha}_i - \frac{\mathbf{a}_i^\top \mathbf{e}}{\rho} \right) = \bar{f} - \frac{\hat{\beta}^2}{\rho}\sum_{i=1}^{n} \Delta_i,$$

which on application of (18) and (16) yield

$$|f(\hat{\alpha}; \mathbf{K}) - \bar{f}| \leq \hat{\beta}\sum_{i=1}^{n} \left| \frac{\hat{\beta}}{\rho}\Delta_i \right| \leq n\hat{\beta}\sqrt{\frac{\log n}{np}} = \left(2(1+\delta)\sqrt{\frac{(1-p)}{p}} \right) \sqrt{\frac{\log n}{p}}. \tag{19}$$

As noted before the function $f(\alpha; \mathbf{K})$ is strongly concave with $\nabla^2_\alpha f(\alpha; \mathbf{K}) \preceq -\frac{\delta}{1+\delta}\mathbf{I}$ for all feasible α. Recalling a useful result from convex optimization, see Lemma 17, we obtain

$$\omega(\mathbf{K}) - f(\hat{\alpha}; \mathbf{K}) \leq \frac{1}{2}\left(1 + \frac{1}{\delta} \right) \|\nabla f(\hat{\alpha}; \mathbf{K})\|^2. \tag{20}$$

Observing that $\nabla f(\alpha;\mathbf{K}) = 2(\mathbf{e}-\alpha-\frac{\mathbf{A}}{\rho}\alpha)$ and using the relation between $\|\cdot\|_\infty$ and $\|.\|$ along with (18) and (17) gives $\|\nabla f(\hat{\alpha};\mathbf{K})\| \leq \sqrt{n}\|\nabla f(\hat{\alpha};\mathbf{K})\|_\infty \leq 2\sqrt{\frac{\log n}{p}}$. Plugging this estimate in (20) and using equation (15) along with (19) we obtain

$$\omega(\mathbf{K}) \leq n\hat{\beta} + \left(1+\frac{1}{\delta}\right)\left(\frac{\log n}{p}\right) + 2(1+\delta)\sqrt{\frac{1-p}{p}}\sqrt{\frac{\log n}{p}}$$

$$= 2(1+\delta)\sqrt{\frac{n(1-p)}{p}} + O\left(\frac{\log n}{p}\right).$$

The second inequality is true because δ is constant. One notes that

$$\text{for any } p(1-p) \geq \frac{(\log n)^2}{n},\ \left(\frac{\log n}{p}\right) \leq \sqrt{\frac{n(1-p)}{p}} \tag{21}$$

holds. The result follows by dividing the first inequality by p^2 and taking square roots. By choice of p as stated in the theorem and for large enough n, one obtains

$$\omega(\mathbf{K}) \leq (1+O(1))\sqrt{\frac{n(1-p)}{p}}$$

and the theorem follows from Remark 18. ■

In the next section we show that when a large independent set is hidden in a random graph one can still view it as a $\mathbf{SVM}-\vartheta$ graph. This property can be very useful in detecting planted cliques in dense graphs.

4.2 Finding Planted Cliques in Random Graphs

Finding large cliques or independent sets is a computationally difficult problem even in random graphs. While it is known that the size of the largest clique or independent set in $G(n,1/2)$ is $2\log n$ with high probability, there is no known efficient algorithm to find a clique of size significantly larger than $\log n$ - even a cryptographic application was suggested based on this (see the discussion and references in the introduction of Feige and Krauthgamer, 2000). Motivated by this, the following problem was introduced (Jerrum, 1992; Kucera, 1995).

Definition 20 (Hidden Planted Clique) *A random $G(n,q)$ graph is chosen first and then a clique of size k is introduced in the first $1,\ldots,k$ vertices. The problem is to identify the clique.*

The case of $q=\frac{1}{2}$ is extensively studied in the literature. Kucera (1995) observed that if $k = \Omega(\sqrt{n\log n})$, then the hidden clique can be easily discovered by examining the high degree vertices. Alon et al. (1998) and Feige and Krauthgamer (2000) showed that if $k=\Omega(\sqrt{n})$, then the hidden clique can be discovered in polynomial time. No efficient algorithm is known to discover the hidden clique if $k=o(\sqrt{n})$.

We consider the (equivalent) complement model $\bar{G}(n,1-p,k)$ where an independent set is planted on the first k vertices and apply the SVM based approach. We show that

Theorem 21 *For $p(1-p) = \Omega(n^{-1}\log^4 n)$, the graph $G = \bar{G}(n,\ 1-p,\ k)$ is a Q graph almost surely if $k = \Omega\left(n^{2/3}p^{-1/3}\ln^{1/3} n\right)$.*

Proof See Appendix B.1 ■

The result shows that the SVM based formulation yields an integral solution for $k \geq c \cdot n^{2/3}\ln^{1/3} n$. More precisely the optimal α for a given G has the property that $\alpha_i = 1$ whenever $i \in \{1,\ldots,k\}$ otherwise $\alpha_i = 0$ whenever k is in the stated regime. Unfortunately this is interesting but not very competitive with state of the art. For $p = \frac{1}{2}$ in the regime, $k = \Omega(\sqrt{n\log n})$, already the highest degree vertices form a clique in the complement $G(n,1/2,k)$ (Kucera, 1995). However, we show next that in the regime $k = \Theta\left(\sqrt{\frac{n(1-p)}{p}}\right)$, the graph $\bar{G}(n,1-p,k)$ is a $\mathbf{SVM}-\vartheta$ graph. Moreover, we can identify the hidden independent set with high probability. In contrast to Feige and Krauthgamer (2000), our algorithm does not involve a SDP and hence will scale to large problems. In Feige and Krauthgamer (2000) the case of $p = 1/2$ was studied and it was conjectured that the results could possibly be extended for a general $p \leq \frac{1}{2}$. In this paper we establish that for large planted clique $\bar{G}(n,1-p,k)$ is indeed a $\mathbf{SVM}-\vartheta$ graph. Moreover the proof motivates an algorithm capable of recovering the clique.

To begin the investigation we will need to compute the ϑ function for $\bar{G}(n,1-p,k)$. For $p = 1/2$, it was shown that the ϑ function is k (Feige and Krauthgamer, 2000). For a general p the ϑ function is also the same as k. To this end we present the following theorem.

Theorem 22 *Let $G = \bar{G}(n,1-p,k)$ where p satisfies $p(1-p) = \Omega(n^{-1}\log^4 n)$. If*

$$k \geq 2\sqrt{\frac{(1-p)}{p}n}(1+o(1)),$$

then, with high probability, $\vartheta(G) = k$.

Proof See Appendix B.2. ■

We study the planted clique problem where k is in the above regime. Indeed for such a choice of k, the graph $\bar{G}(n,1-p,k)$ is a $\mathbf{SVM}-\vartheta$ graph. One of the main contribution of this paper is the following theorem.

Theorem 23 *Let $p(1-p) = \Omega(n^{-1}\log^4 n)$ and $\delta > 0$ be a given constant. For any $G = \bar{G}(n,1-p,k)$ and $k = 2t\sqrt{\frac{n(1-p)}{p}}$, for large enough constant $t \geq 1$ with $\mathbf{K}$ as in (2) and*

$$\rho = \left(2\sqrt{np(1-p)} + kp\right)(1+\delta),$$

the following holds:

$$\omega(\mathbf{K}) \leq \left(1 + \frac{1}{t}(1+\delta) + \delta\right)\vartheta(G) + o(1), \tag{22}$$

with probability at least $1 - O(1/n)$.

Note that we need a stronger regime for p in the above theorem when compared to Theorem 19. This is necessary in view of the conditions in Theorem 21 and Theorem 22.

As a preliminary, we need a bound on the minimum eigenvalue of $\bar{G}(n,1-p,k)$:

Lemma 24 *With high probability, the minimum eigenvalue of $\bar{G}(n,1-p,k)$ is bounded in absolute value by $2\sqrt{np(1-p)}+kp$.*

Proof We write the adjacency matrix $\mathbf{A}$ of $\bar{G}(n,p,k)$ as $\mathbf{A}' = \mathbf{A}+\mathbf{E}$ where $\mathbf{A}'$ is the adjacency matrix of $G(n,1-p)$ and observe that E is zero except for a $k\times k$ block where it is the adjacency matrix of $G(k,1-p)$. Using the Weyl perturbation inequality (Horn and Johnson, 1990), we deduce that $\lambda_n(\mathbf{A}) \geq \lambda_n(\mathbf{A}') - \lambda_1(\mathbf{E})$. By Remark 18 we see that with high probability $\lambda_n(\mathbf{A}') \geq -2\sqrt{np(1-p)}$ and $\lambda_1(\mathbf{E}) = kp$ ■

Proof (of Theorem 23) The proof is analogous to that of Theorem 19. Let $\mathbf{A}$ be the adjacency matrix of G. By definition of ρ and by Lemma 24 we see that $\rho = (1+\delta)|\lambda_n(\mathbf{A})|$. For any $\delta > 0$, we see that the minimum eigenvalue of $\frac{1}{\rho}\mathbf{A}+\mathbf{I}$ is almost surely *strictly* greater than 0. As a consequence the function $f(\alpha,\mathbf{K})$ (see (1)) is strongly concave for any instance of G. We begin by analysing the KKT conditions, which are necessary and sufficient for optimising strongly concave problems. The KKT conditions in (3) specialises to the graph G as follows

$$\begin{array}{rcl} \alpha_i + \frac{1}{\rho}\sum_{j=k+1}^{n} A_{ij}\alpha_j & = & 1+\mu_i, \quad \mu_i\alpha_i = 0, \quad \mu_i \geq 0 \quad \forall 1 \leq i \leq k; \quad \text{and} \\ \alpha_i + \frac{1}{\rho}\sum_{j=1}^{n} A_{ij}\alpha_j & = & 1+\mu_i, \quad \mu_i\alpha_i = 0, \quad \mu_i \geq 0 \quad \forall k+1 \leq i \leq n. \end{array} \tag{23}$$

Let us analyse the average case. By the conditions of the theorem, the expectation of $\mathbf{A}$ is defined as follows

$$\mathbb{E}(\mathbf{A})_{ij} = \begin{cases} 0 & \text{whenever } 1 \leq i,j \leq k \text{ and } i = j, \\ p & \text{otherwise.} \end{cases}$$

By direct verification the matrix $\tilde{\mathbf{K}} = \frac{\mathbb{E}(\mathbf{A})}{\rho} + \mathbf{I}$ is seen to be positive definite. More importantly $f(\alpha,\tilde{\mathbf{K}})$ is again strongly concave and attains maximum at a KKT point. A KKT point for the problem

$$\max_{\alpha \geq 0} f(\alpha,\tilde{\mathbf{K}})$$

is given by $\hat{\alpha} = [\beta_1 \mathbf{e}_k^\top \ \beta_2 \mathbf{e}_{(n-k)}^\top]$ where β_1 and β_2 satisfies

$$\beta_1 + \frac{(n-k)p}{\rho}\beta_2 = 1, \quad \text{and,} \quad \beta_2 + \frac{k}{\rho}p\beta_1 + \frac{(n-k-1)}{\rho}p\beta_2 = 1;$$

which leads to

$$\beta_1 = \frac{1}{1+\frac{np}{\rho}(1-r)}, \quad \text{and,} \quad \beta_2 = \frac{1}{\frac{1}{1-r}+\frac{np}{\rho}}.$$

For k and ρ as given, the value of r is defined as

$$r = \frac{k}{\rho}p = \frac{t}{(1+t)(1+\delta)}, \qquad 1-r \approx \frac{1}{t+1}.$$

For small δ, the second approximation holds. Further noting that $k = o(n)$ allows us to write

$$\beta_1 = \frac{(t+1)^2}{(t+1)^2 + \frac{1}{2}\sqrt{\frac{np}{1-p}}\frac{1}{1+\delta}} \qquad \text{and} \qquad \beta_2 = \frac{(t+1)}{(t+1)+\frac{1}{2}\sqrt{\frac{np}{1-p}}\frac{1}{1+\delta}}. \tag{24}$$

By construction $\hat{\alpha}$ is the optimal solution for the expected case and the optimal value, $\bar{f}$ is given by

$$\bar{f} = k\beta_1 + (n-k)\beta_2 \;\leq\; o(k) + 2\frac{n(t+1)(1+\delta)}{\sqrt{\frac{np}{1-p}}} \;=\; \left(1+\delta+\frac{1}{t}(1+\delta)+o(1)\right)k. \qquad (25)$$

As $\beta_1 < 1$, the first term is $o(k)$. The second inequality follows by noting that second term can be upper-bounded by neglecting $(t+1)^2$ in the denominator of β_2. The last equality follows from definition of k. Now as in the case of $G(n,p)$ graphs, we can indeed show that $\hat{\alpha}$ is an approximate KKT solution for a random graph G.

Let $deg(i,T)$ = number of edges from i to T where $T \subset V$. Since $deg(j,S) \sim Bin(k,p)$ for any $j \in \bar{S}(= V \backslash S)$ random variable, we can deduce that

$$|deg(j,S) - kp| \leq \sqrt{kp\log k} \text{ with probability at least } 1 - O(1/k)$$

by application of Chernoff bound. Similarly the following is true for any $j \in V$

$$|deg(j,\bar{S}) - (n-k)p| \leq \sqrt{(n-k)p\log(n-k)}$$

with high probability. Using in $\hat{\alpha}$ in (23) we obtain

$$\begin{aligned}
\forall i \in S \qquad & |\beta_1 + \tfrac{deg(i,\bar{S})}{\rho}\beta_2 - 1| \leq && \frac{\sqrt{(n-k)p\log(n-k)}}{\rho}\beta_2, \quad \text{and} \\
\forall i \in \bar{S} \quad & |\beta_2 + \tfrac{deg(i,S)}{\rho}\beta_1 + \tfrac{deg(i,\bar{S})}{\rho} - 1| \leq && \frac{\sqrt{kp\log k}}{\rho}\beta_1 + \frac{\sqrt{(n-k)p\log(n-k)}}{\rho}\beta_2.
\end{aligned}$$

Plugging in the values of β_1 and β_2, see (24), gives us the following claim.

Lemma 25 *Let β_1 and β_2 be defined in* (24)*. The vector $\bar{\alpha}^\top = [\beta_1 \mathbf{e}_k^\top \; \beta_2 \mathbf{e}_{n-k}^\top]$ satisfies the following relationship with $\varepsilon = O\left(\sqrt{\frac{\log n}{np}}\right)$.*

$$\begin{aligned}
1 \leq i \leq k \qquad & |\hat{\alpha}_i + \frac{1}{\rho}\sum_{j=k+1}^{n} A_{ij}\hat{\alpha}_j - 1| \leq \varepsilon, && \textit{and} \\
k+1 \leq i \leq n \qquad & |\hat{\alpha}_i + \frac{1}{\rho}\sum_{j=1}^{n} A_{ij}\hat{\alpha}_j - 1| \leq \varepsilon.
\end{aligned}$$

Proof By direct verification. ■

The above lemma establishes that $\hat{\alpha}$ is an ε-approximate KKT point. Indeed

$$|f(\hat{\alpha};\mathbf{K}) - \bar{f}| \leq (k\beta_1 + (n-k)\beta_2)\varepsilon = O(k\varepsilon) = O\left(\frac{1}{\sqrt{p}}\sqrt{\frac{(1-p)}{p}}\sqrt{\log n}\right). \qquad (26)$$

The first equality follows by using Lemma 25, and the second equality follows from (25). The last equality follows from definition of k and ε. Again using the strong concavity condition $\nabla^2_\alpha f(\alpha;\mathbf{K}) \preceq -\frac{\delta}{1+\delta}\mathbf{I}$ for all feasible α and using Lemma 17, we obtain

$$\omega(\mathbf{K}) - f(\hat{\alpha};\mathbf{K}) \leq \frac{1}{2}\left(1+\frac{1}{\delta}\right)\|\nabla f(\hat{\alpha};\mathbf{K})\|^2. \qquad (27)$$

Algorithm 2 Input: $\mathbf{A}(\bar{G}(n,1-p,k)),k$ Output: $S \subset \{1,\ldots,n\}, |S| = k$

$\alpha^* =$ Use SVM solver to solve (1) with $\mathbf{K}$ as defined in Theorem 23

Return $S = \{i_j | j = 1,\ldots,k\ \alpha^*_{i_1} \geq \alpha^*_{i_2} \ldots \geq \alpha^*_{i_n}\}$

Recalling that $\nabla f(\alpha;\mathbf{K}) = 2(\mathbf{e} - \alpha - \frac{\mathbf{A}}{\rho}\alpha)$ and using the relation between $\|\cdot\|_\infty$ and 2 norm along with Lemma 25 gives $\|\nabla f(\hat{\alpha};\mathbf{K})\| \leq \sqrt{n}\|\nabla f(\hat{\alpha};\mathbf{K})\|_\infty \leq O(\sqrt{\frac{\log n}{p}})$. Plugging this estimate in (27) and using equation (26) we obtain

$$\omega(\mathbf{K}) \leq \hat{f} + O\left(\frac{\log n}{p}\right) + O\left(\frac{1}{\sqrt{p}}\sqrt{\frac{(1-p)}{p}}\sqrt{\log n}\right).$$

For the given choice of p one can use (21) to write

$$\omega(\mathbf{K}) \leq \left(1 + \frac{1}{t}(1+\delta) + \delta + o(1)\right)k.$$

By Theorem 22, $\vartheta(\bar{G}(n,1-p,k) = k$ holds with high probability, proving (22). ■

The above theorem establishes that $\hat{\alpha}$ is a very good approximation for the optimal solution with high probability.

We note that $\hat{\alpha}_i$, for any vertex i in the independent set $S = \{1,\ldots,k\}$, is $(t+1)$ times larger than any other $\hat{\alpha}_j \notin S$. Since α^* is very close to $\hat{\alpha}$ one might expect to see the same property in α^*. This motivates Algorithm 2 for finding planted clique in a random graph.

It is expected that with high probability the largest k values in the optimal solution would correspond to nodes in the independent set. In a later section we will present empirical results which show that the algorithm is extremely successful in recovering the clique. The runtime of this algorithm is dependent on the availability of an efficient SVM solver. Current SVM solvers have complexity less than $O(n^2)$, hence the proposed procedure is indeed scalable to large graphs. In contrast the best known algorithm for solving Lovász ϑ has a runtime complexity of $O(n^5 \log n)$ (Chan et al., 2009). The algorithm in Feige and Krauthgamer (2000) is based on computation of the ϑ function and hence will not scale well to large graphs.

4.3 Finding Planted Subgraphs in Random Graphs

The above results show that the SVM procedure can recover a planted independent set in a sparse random graph, which is later exploited to solve the planted clique problem in a dense graph. Moreover we have also established that on random graphs the SVM objective function is tightly concentrated. This points to a more general question that of identifying a planted subgraph, which is much sparser than the original graph.

Let $G(n,p,p',k)$ be a graph which has a planted $G(k,p')$ graph on the first k vertices of a random $G(n,p)$ graph. We consider the problem of recovering the planted subgraph.

An interesting property of ϑ function is that deletion of edges always results in an increases of the ϑ function. As a consequence for any k

$$\vartheta(\bar{G}(n,1-p,k)) \geq \vartheta(G(n,p,p',k)) \geq \vartheta(G(n,p))$$

whenever $p' < p$. By Remark 18 and by Theorem 22 we have $\vartheta(\bar{G}(n,1-p,k))$ and $\vartheta(G(n,p))$ are both $\Theta\left(\sqrt{\frac{n(1-p)}{p}}\right)$ whenever $k = \Theta\left(\sqrt{\frac{n(1-p)}{p}}\right)$. Indeed one can show that the graph with a planted subgraph is also a $\mathbf{SVM}-\vartheta$ graph. More formally

Theorem 26 *Let $G = G(n,p,p',k)$ with $p(1-p) = \Omega(n^{-1}\log^4 n)$ and $p' < p$. If*

$$k = 2t\left(\sqrt{\frac{n(1-p)}{p}}\frac{1-2p'}{1-\frac{p'}{p}}\right),$$

then the graph G satisfies

$$\omega(\mathbf{K}) \leq \left(1+\frac{1}{t}(1+\delta)\right)\vartheta(G),$$

with probability at least $1-O(1/n)$ where $\mathbf{K}$ as in (2) and $\rho = \left(2\sqrt{np(1-p)}+k(p-p')\right)(1+\delta)$ for any constant $\delta > 0$.

The proof can be constructed as in Theorem 23 so we do not repeat it here. Instead we make some observations which would help to construct a formal proof by repeating the arguments in Theorem 23. One would need to compute an upper-bound on $\lambda_n(\mathbf{A})$ where $\mathbf{A}$ is the adjacency matrix of $G(n,p,p',k)$. We write $\mathbf{A}' = \mathbf{A}+\mathbf{E}$ where $\mathbf{A}'$ is the adjacency matrix of $G(n,p)$ graph. The matrix $\mathbf{E}$ is defined as follows. Set all entries of $\mathbf{E}$ to be 0 except the first $k \times k$ submatrix. On that block set

$$E_{ij} = 1 \text{ with probability } s = \tfrac{p-p'}{1-2p'} \text{ whenever } A_{ij} = 0 \text{ for } 1 \leq i,j \leq k,$$

otherwise $E_{ij} = 0$. By such a choice we see that $\mathbf{E}$ is the adjacency matrix of $G(k,s)$ on the first $k \times k$ block and 0 everywhere else. To check if $P(A'_{ij} = 1) = p$ we note that all entries, except in the leading $k \times k$ submatrix, are $Ber(p)$ distributed. In the leading $k \times k$ submatrix we note that $A'_{ij} = 1$ is the union of two mutually exclusive events, namely $A_{ij} = 1, E_{ij} = 0$ or $A_{ij} = 0, E_{ij} = 1$. Substituting the value of s confirms

$$P(A'_{ij} = 1) = P(A_{ij} = 1)P(E_{ij} = 0) + P(A_{ij} = 0)P(E_{ij} = 1) = p'(1-s) + (1-p')s = p.$$

Now by Weyl Perturbation (Horn and Johnson, 1990) we can write

$$\lambda_n(\mathbf{A}) \geq \lambda_n(\mathbf{A}') - \lambda_1(\mathbf{E}),$$

which leads to $|\lambda_n(\mathbf{A})| \leq 2\sqrt{np(1-p)} + k\frac{p-p'}{1-2p'}$. This explains the choice of ρ. As before the expected case analysis leads to the following equation

$$\beta_1 + \frac{kp'}{\rho}\beta_1 + \frac{(n-k)p}{\rho}\beta_2 = 1 \quad \text{and} \quad \beta_2 + \frac{kp}{\rho}\beta_1 + \frac{(n-k-1)p}{\rho}\beta_2 = 1.$$

The theorem follows by retracing the steps in the proof of Theorem 23. As in the planted clique case the Algorithm 2 recovers the planted subgraph. The proofs in this section suggest that $\omega(\mathbf{K})$ maybe concentrated. Before we close the section we would like to show that indeed this is the case.

4.4 Concentration of $\omega(\mathbf{K})$

The SVM objective function, computed on the random graphs in the previous sections, is also highly concentrated. Using the celebrated Talagrand's inequality (Talagrand, 1995), one can prove the following theorem

Theorem 27 *Let $\omega(\mathbf{K})$ be defined as in* (1)*. Let G and ρ satisfy either of*

- $G = G(n, p)$, $\rho = (1+\delta)2\sqrt{np(1-p)}$
- $G = \bar{G}(n, 1-p, k)$, $\rho = (1+\delta)(2\sqrt{np(1-p)}+kp)$ *where* $k = 2t\sqrt{\frac{n(1-p)}{p}}$,

where $p(1-p) = \Omega(n^{-1}\log^2 n)$. Then there exists a constant M such that

$$|\omega(\mathbf{K})) - M| \leq O\sqrt{\frac{\ln^3 n}{np^3(1-p)}}$$

with probability $1 - O(\frac{1}{n})$ whenever $\mathbf{K}$ is defined in (2)*.*

Before we begin the proof we collect some results. Let $\omega(\mathbf{K}) = f(\alpha^*, \mathbf{K})$, see (1). For any graph G satisfying the conditions of the theorem we can establish that

Lemma 28 *For α^* be defined as above we have $\|\alpha^*\| = O\left(\sqrt{\frac{\log n}{p}}\right)$ with probability $1 - \frac{1}{O(n)}$*

Proof For $G(n,p)$ case, application of Lemma 17 allows us to rewrite (20)

$$\frac{1}{2}\frac{\delta}{1+\delta}\|\hat{\alpha} - \alpha^*\|^2 \leq \omega(\mathbf{K}) - f(\hat{\alpha}; \mathbf{K}) \leq \frac{1}{2}\left(1 + \frac{1}{\delta}\right)\|\nabla f(\hat{\alpha}; \mathbf{K})\|^2 \leq O\left(\sqrt{\frac{\log n}{p}}\right)$$

where $\hat{\alpha}$ satisfies (14). Note that this holds with probability at least $1 - \frac{1}{O(n)}$ This yields the result $\|\hat{\alpha} - \alpha^*\| \leq O\left(\sqrt{\frac{\log n}{p}}\right)$ and the lemma follows by noting that $\|\hat{\alpha}\| = O(1)$. Similarly for $G = G(n, 1-p, k)$, application of Lemma 17 to (27) yields the lemma. ■

We employ the following version of Talagrand's inequality

Theorem 29 (Dubhashi and Panconesi, 2009) *Let $x = (x_1, \ldots, x_d)$ where $x_1, \ldots, x_d$ are independent random variables with $x_i \in \Omega_i$. Let $f : \Omega = (\Omega_1 \times \ldots \times \Omega_d) \to \mathbb{R}$ be a function with the property that if for each x in the domain of f, there exists a vector $c = c(x)$ such that $\|c\| \leq B$ and*

$$f(x) \leq f(y) + \sum_{x_i \neq y_i} c_i, \ \forall\, y \in \Omega,$$

then for any $s > 0$:

$$Pr[|f(x) - M| > Bs] \leq 4e^{-s^2/4},$$

where M denotes the median of f.

Proof See Theorem 11.2 in Dubhashi and Panconesi (2009). ■

The application of the above theorem establishes the result

Lemma 30 *Let* $\mathbf{A}$ *be the adjacency matrix of G, as defined in Theorem 27. Let* α^* *denote the solution to the optimization problem*

$$v(\mathbf{A}) = \max_{\alpha \geq 0} 2e^T\alpha - \alpha^T\left(I + \frac{\mathbf{A}}{\rho}\right)\alpha \tag{28}$$

for some constant ρ*. Then for any* $b > 0$*, there exists a constant* M *such that*

$$Pr\left[|v(\mathbf{A}) - M| > \frac{\sqrt{2}b^2 s}{\rho}\right] \leq 4e^{-s^2/4} + Pr[\|\alpha^*\| > b] \tag{29}$$

for any $s > 0$*.*

Proof Let β^* be the optimal solution of

$$f(\mathbf{A}) := \max_{\beta \geq 0, \|\beta\| \leq b} 2e^T\beta - \beta^T\left(I + \frac{\mathbf{A}}{\rho}\right)\beta.$$

Since (28) can be seen as a relaxation of this, it follows that if α^* is feasible, that is, $\|\alpha^*\| \leq b$, then $v(\mathbf{A}) = f(\mathbf{A})$. For any $D > 0$, this means that if for some M, the inequality $|v(\mathbf{A}) - M| > D$ holds then either $|f(\mathbf{A}) - M| > D$ or $\|\alpha^*\| > b$, so

$$Pr[|v(\mathbf{A}) - M| > D] \leq Pr[|f(\mathbf{A}) - M| > D] + Pr[\|\alpha^*\| > b]. \tag{30}$$

By the definition of f we know that for any adjacency matrix $\mathbf{A}'$

$$\begin{aligned} f(\mathbf{A}') &\geq 2e^T\beta^* - (\beta^*)^T\left(\mathbf{I} + \frac{\mathbf{A}'}{\rho}\right)\beta^* \\ &= f(\mathbf{A}) + \frac{1}{\rho}\sum_{i,j}(A_{ij} - A'_{ij})\beta_i^*\beta_j^* \\ &\geq f(\mathbf{A}) - \frac{1}{\rho}\sum_{A_{ij} \neq A'_{ij}} 2\beta_i^*\beta_j^*, \end{aligned}$$

where the last sum goes over all $i > j$. Let $c_{ij} = \frac{2\beta_i^*\beta_j^*}{\rho}$ for $i > j$. We note that

$$\sum_{i>j} c_{ij}^2 \leq 2\frac{1}{\rho^2}\sum_{i,j}(\beta_i^*)^2(\beta_j^*)^2 = \frac{2\|\beta^*\|^4}{\rho^2} \leq \frac{2b^4}{\rho^2}.$$

The function f is defined on $A_{ij}, i < j$ a total of $d = \binom{n}{2}$ Bernoulli random variables for the $G(n,p)$ case. For the planted clique case one needs to consider $d = \binom{n}{2} - \binom{k}{2}$ random variables. Applying Theorem 29 on f with $c_{ij} = \frac{2\beta_i^*\beta_j^*}{\rho}$ for $i > j$ and $B = \sqrt{2}b^2/\rho$ we get that there exists an M such that for any $s > 0$

$$Pr\left[|f(\mathbf{A}) - M| > \frac{\sqrt{2}b^2 s}{\rho}\right] \leq 4e^{-s^2/4}.$$

Using this bound together with (30) where $D = \sqrt{2}b^2 s/\rho$ implies (29). ■

Proof [Proof of Theorem 27.] By Lemma 28 With probability $1 - \frac{1}{O(n)}$, for either one of $G = G(n, p)$ or $G = \bar{G}(n, 1-p, k)$, there exists a constant $C > 0$ such that $\|\alpha^*\| \leq C\sqrt{\frac{\ln n}{p}}$. Applying Lemma 30 with $b = C\sqrt{\frac{\ln n}{p}}$ and $s = 2\sqrt{\ln n}$ yields

$$Pr\left[|v(\mathbf{A}) - M| > C'\frac{\sqrt{\ln^3 n}}{\rho p}\right] = O\left(\frac{1}{n}\right),$$

for some constant $C' > 0$. By assumption $\rho = \Theta\left(\sqrt{np(1-p)}\right)$, and hence the above equation establishes that $|v(\mathbf{A}) - M| \leq O\sqrt{\frac{\ln^3 n}{np^3(1-p)}}$ with probability $1 - O(\frac{1}{n})$. ■

5. Experimental Evaluation

In Section 3 an algorithm was proposed which was capable of discovering a large common dense subgraph in a collection of graphs. In Section 4.2 an algorithm for discovering a large planted clique in a single graph was discussed. In this section we examine the performance of the proposed algorithms. The code and data for the experiments reported here is available online at `http://www.cse.chalmers.se/~jethava/svm-theta.html`.

5.1 Common Dense Subgraph Detection

This subsection presents an experimental evaluation of the *CSS* algorithm for finding large dense subgraphs across multiple graphs. We study an abstraction of the problem using the DIMACS'94 data set which consists of graphs motivated from a number of different practical problems (Johnson and Trick, 1996). The key property of the algorithm is that it is parameter-less, and that it finds large common dense regions - which was not possible using enumerative approach (Jiang and Pei, 2009). We also investigate a thresholding heuristic which improves induced subgraph density at the cost of subgraph size.

5.1.1 DATA SET

We evaluate our algorithm for finding large dense regions on the DIMACS Challenge graphs,[2] which is a comprehensive benchmark for testing of clique finding and related algorithms. Each of the graph families in DIMACS (*brock*, *c-fat*, *p_hat*, *san*, *sanr*) is motivated by carefully selected real world problems, for example, fault diagnosis (*c-fat*), etc.; thus covering a wide range of practical scenarios (Johnson and Trick, 1996).

We evaluate the algorithm on following class of graphs *c-fat* graphs which are based on fault diagnosis problems and are relatively sparse; and, *p_hat* graphs which are generalizations of Erdös-Rényi random graphs; and are characerized by wider degree spread compared to classical $G(n, p)$

2. The DIMACS benchmark data set is available online at `ftp://dimacs.rutgers.edu/pub/challenge/graph/benchmarks/clique/`.

model. For the families of dense graphs (*brock*, *san*, *sanr*), we focus on finding large dense region in the complement of the original graphs.

5.1.2 EVALUATION

We run Algorithm 1 using off-the-shelf MKL solver SimpleMKL,[3] to find large common dense subgraph. In order to evaluate the performance of our algorithm, we compute $\bar{a} = \max_l a^{(l)}$ and $\underline{a} = \min_l a^{(l)}$ where $a^{(l)} = \gamma(G_T^{(l)})/\gamma(G^{(l)})$ is relative density of induced subgraph (compared to original graph density); and n_T/N is relative size of induced subgraph compared to original graph size. We want a high value of n_T/N; while $\underline{a}$ should not be lower than 1.

Graph family	N	M	n_{SV}	n_T	$\bar{a}$	$\underline{a}$
c-fat200	200	3	100	90	**2.3613**	0.99165
c-fat500	500	4	152	140	**3.8846**	1.0182
brock200‡	200	4	165	164	1.0704	0.9954
brock400‡	400	4	397	349	1.0214	1.0084
brock800‡	800	4	795	686	1.0129	1.0062
p_hat300	300	3	158	157	**1.5296**	1.1456
p_hat500	500	3	239	238	**1.5551**	1.1722
p_hat700	700	3	313	312	**1.5782**	1.1818
p_hat1000	1000	3	429	428	**1.5976**	1.1879
p_hat1500	1500	3	574	573	**1.6313**	1.2011
san200‡	200	5	200	185	1.0458	1.0029
san400‡	400	3	359	358	1.0411	0.9947
sanr200‡	200	2	157	156	1.1402	0.9949
sanr400‡	400	2	343	342	1.0355	1.0017

Table 1: Common dense subgraph recovery on multiple graphs in DIMACS data set. Here $\bar{a}$ and $\underline{a}$ denote the maximum and minimum relative density of the induced subgraph (relative to density of the original graph); n_{SV} and n_T denotes the number of support vectors and size of subset $T \subseteq SV$ returned by Algorithm 1. *The enumerative algorithm does not run for large dense regions* (see Jiang and Pei, 2009, Figure 17).

Table 1 shows evaluation of Algorithm 1 on DIMACS data set. We note that our algorithm finds a large subgraph (large n_T/N) with higher density compared to original graph in all of DIMACS graph classes making it suitable for finding large dense regions in multiple graphs. We note that traditional set enumerative methods fail to handle dense subgraph recovery for the case when n_T/N is large. For example, finding quasicliques of size $n_T \simeq 60$ requires 11.5 hours (see Jiang and Pei, 2009, Figure 17); in contrast to MKL-based approach which takes less than 1 minute.

The results in Table 1 show that in case of *c-fat* and *p_hat* graph families, the induced subgraph density is significantly improved (evidenced by high $\bar{a}$ and $\underline{a}$); and, the number of support vectors n_{SV} is a large fraction of N ($n_{SV}/N \simeq 1/2$). Thus, the algorithm recovers a large common dense subgraph.

3. The SimpleMKL solver is available online at `http://asi.insa-rouen.fr/enseignants/~arakotom/code/mklindex.html`.

Graph family	N	M	n_T	n_{S_c}	$\bar{a}(T)$	$\underline{a}(T)$	$\bar{a}(S_c)$	$\underline{a}(S_c)$
brock200‡	200	4	164	83	1.0704	0.9954	1.36	0.99
brock400‡	400	4	349	199	1.0214	1.0084	1.15	1.05
brock800‡	800	4	686	398	1.0129	1.0062	1.08	1.01
san200‡	200	5	185	100	1.0458	1.0029	1.51	1.08
san400‡	400	3	358	180	1.0411	0.9947	1.19	1.02
sanr200‡	200	2	156	79	1.1402	0.9949	1.86	1.04
sanr400‡	400	2	342	172	1.0355	1.0017	1.20	1.02

Table 2: Heuristic rule: If $n_{SV} \geq 0.8n$ then choose c such that $|S_c| = n_{SV}/2$. Here $\bar{a}$ and $\underline{a}$ denote the maximum and minimum relative density of the induced subgraph (relative to density of the original graph). Induced subgraph density improves by choosing S_c instead of T, that is, $\bar{a}(S_c) \geq \bar{a}(T)$ and $\underline{a}(S_c) \geq \underline{a}(T)$ for all graphs.

On the other hand, for *brock* and *san* graph families, the number of support vectors is equal to the overall graph size $n_{SV} \simeq N$; and consequently the relative density is 1, that is, $\gamma(G^{(l)}_{SV}) = \gamma(G^{(l)})$ which is not interesting. In the following subsection, we discuss a heuristic for improving subgraph detection performance when all nodes are support vectors.

5.2 Thresholding Heuristic

In this section, we discuss the impact of choosing support vectors with *high* support by choosing some $c > 0$ and selecting the set $S_c = \{i : \alpha^*_i > c\}$. Figure 2 shows the densities of the induced subgraph $\gamma(G^{(l)}_{S_c})$ relative to original graph density $\gamma(G^{(l)})$ for all graphs $G^{(l)} \in \mathbb{G}$, that is,

$$a^{(l)} = \gamma(G^{(l)}_{S_c})/\gamma(G^{(l)}) \forall\, G^{(l)} \in \mathbb{G}$$

at different c thresholds varying between $c = 0$ ($|S_c| = |SV|$) and $c = 1$ ($|S_c| = 0$) (and correspondingly different subgraph sizes $|S_c|$ which is shown on x-axis).

Figure 2 shows the variation in density of the induced subgraph $\gamma(G^{(l)}_{S_c})$ relative to original graph density $\gamma(G^{(l)})$ for all graphs $G^{(l)} \in \mathbb{G}$ at increasing subgraph sizes for the largest graph (resp. *c-fat500, p_hat1500, brock800, san400* and *sanr400*) in each graph family (resp. *c-fat, p_hat, brock, san* and *sanr*). Figure 5 in Appendix A presents the variation for other DIMACS graphs.

Notice that in case of *c-fat* and *p_hat* graph families (Figures 2(a) and 2(c)), one can further improve graph densities across all graphs $G^{(l)} \in \mathbb{G}$ by choosing a higher value of c (and correspondingly, a smaller induced subgraph $|S_c|$). In the remaining examples, choosing a higher value of c (and correspondingly lower $|S_c|$) improves the density in at least one $G^{(l)} \in \mathbb{G}$.

Based on performance on the DIMACS data set, we suggest a simple rule for using the heuristic whenever $n_{SV} \geq 0.8n$, then we choose c such that $|S_c| = \lceil n_{SV}/2 \rceil$. Table 2 shows the improvement by using the heuristic rule whenever $n_{SV} \geq 0.8n$. We note that minimum and maximum induced subgraph density improves by choosing S_c instead of T, that is, $\bar{a}(S_c) \geq \bar{a}(T)$ and $\underline{a}(S_c) \geq \underline{a}(T)$ for all graph families.

It can be seen from Figure 2 that the induced subgraph density is not strictly monotonic with induced subgraph size $|S_c|$. However, for medium to large $|S_c|$, the subgraph density shows a general

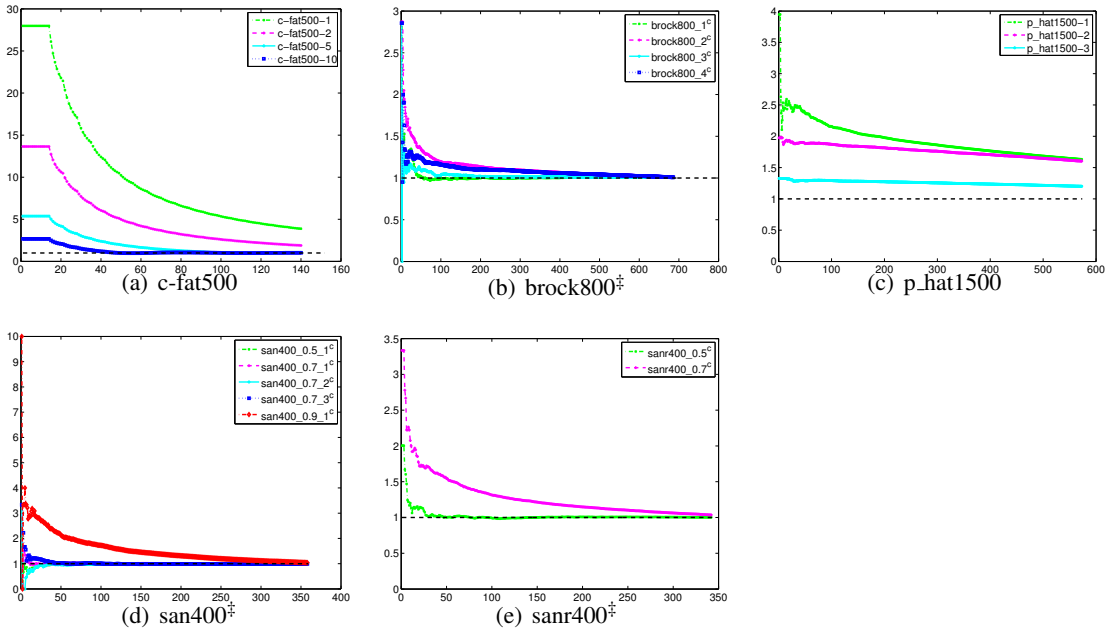


Figure 2: Common dense subgraph recovery. Figures $(a)-(e)$ show the densities of the induced subgraph $\gamma(G^{(l)}_{S_c})$ relative to original graph density $\gamma(G^{(l)})$ for all graphs $G^{(l)} \in \mathbb{G}$ at different values of $c \in [0,1]$ (i.e., different subgraph sizes $|S_c|$) for different DIMACS graph families.

decreasing trend, that is, if $|S_c| > |S_{c'}|$, then the induced subgraph density $\gamma(G^{(l)}_{S_c}) < \gamma(G^{(l)}_{S_{c'}})$ for some regime $c, c' \in [c_l, c_h]$. Thus, one can choose a smaller induced subgraph S_c having higher induced subgraph density by selecting higher value of threshold c.

It is instructive to note that in all graph families, the graph with maximum relative density, for example, *c-fat500* in Figure 2(a) is the graph with minimum average density among all graph $\mathbb{G}$. In other words, *MKL-based approach tries to find a dense region in the sparsest subgraph* $G^{(l)} \in \mathbb{G}$ *while making sure it is compatible with remaining graphs in* $\mathbb{G}$.

5.3 Planted Clique Recovery on Random Graphs

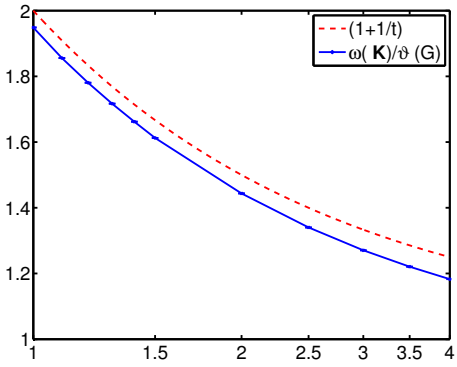
We consider the case for Erdos-Renyi graph with general $p \neq 1/2$ and planted clique of size k, that is, $G(n,p,k)$.

5.3.1 DATA SET

We generate $n_s = 100$ random graphs based on $\bar{G}(n, 1-p, k)$ with planted independent set of size $k = 2t\sqrt{n(1-p)/p}$ and $p = \frac{1}{2}n^{-\alpha}$ where $n = 20000$ and $\alpha \geq 0$. We choose $\alpha = c/30$ where c lies in the set $\{0, 1, 2, \ldots, 10\}$; and perform n_s experiments for each c. Note that the case of $\alpha = 0$ yields the familiar planted clique model $G(n, 1/2, k)$ with $k = 2t\sqrt{n}$.

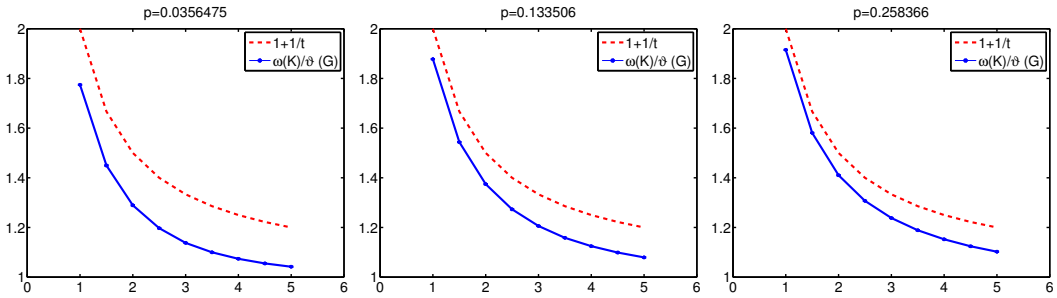
t	f_r	F_1
1	0	0.007
2	1	1
3	1	1
4	1	1

(a)



(b)

Figure 3: (a) shows f_r the fraction of graphs for which the hidden clique is recovered exactly; and, the average F_1-score measuring quality of recovered subset over n_s trials at each t ($k = 2t\sqrt{n}$). (b) shows $\omega(\mathbf{K})/k$ (blue) is bounded by $(1+1/t)$ (red) (Theorem 23). This allows approximation of Lovász ϑ function for large $\mathbf{SVM}-\vartheta$ graphs without SDP.



(a) (b) (c)

Figure 4: Approximation of Lovász ϑ function for general p. This figure shows $\omega(\mathbf{K})/k$ (blue) is bounded by $(1+1/t)$ (red) (Theorem 23). This allows approximation of Lovász ϑ function for large $\mathbf{SVM}-\vartheta$ graphs without SDP for general p.

5.3.2 EVALUATION

We consider the **LS** labelling $\mathbf{K} = \frac{A}{\rho} + I$ of $G(n, 1-p, k)$ for $\rho = 2\sqrt{np(1-p)} + kp$ as discussed in Section 4.2. We solve $\omega(\mathbf{K})$ using libsvm solver,[4] and return the top-k support vectors $\tilde{S}_k$ as independent set as discussed in Algorithm 2.

In order to evaluate algorithm accuracy, we compute the fraction of graphs for which the independent set is recovered exactly using Algorithm 2, that is, $f_r = n_r/n_s$ where n_r is the number of graphs for which independent set is recovered exactly for each $\bar{G}(n, 1-p, k)$. We also compute the average (over n_s trials) *F1* score which measures how different is our solution $\tilde{S}_k$ to maximum independent set S as: $F_1 = \frac{2pr}{p+r}$, $p = \frac{|\tilde{S}_k \cap S|}{|\tilde{S}_k|}$, $r = \frac{|\tilde{S}_k \cap S|}{|S|}$.

4. LibSVM solver is available online at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

5.3.3 DISCUSSION

As predicted by Theorem 23, there exists some $t_0 > 0$ for which Lovász ϑ function is bounded by $\omega(\mathbf{K})$; and the planted clique can be recovered perfectly by selecting the top k support vectors in sorted descending order of α_i^*. We find experimentally that this approach recovers the planted clique *exactly* for $t \geq 2$ for all $c \in \{0,\ldots,10\}$, that is, random graph $\bar{G}(n,1-p,k)$ with $p = \frac{1}{2}n^{-\alpha}$ and planted independent set of size $k = 2t\sqrt{n(1-p)/p}$.

In particular we discuss the case $c = 0$ which yields the Erdös-Rényi graph with $p = 1/2$ and planted clique of size $k = 2t\sqrt{n}$. Figure 3(a) shows the fraction of graphs for which the hidden clique is recovered exactly using above procedure. Figure 3(b) shows $\omega(\mathbf{K})/k$ (shown in blue), and consequently $\omega(\mathbf{K})/\vartheta(G)$ is bounded by $(1+1/t)$ (shown in red) for $G(n,1/2,k)$ case. Figure 4 shows that $\omega(\mathbf{K})/\vartheta(G)$ is bounded by $(1+1/t)$ for case when $p \neq 1/2$. Thus, *one can use* $\omega(\mathbf{K})$ *for approximating Lovász* ϑ *function for large* $\mathbf{SVM}-\vartheta$ *graphs without solving a SDP problem*.

6. Conclusion

The results in this paper establish that there is a close connection between the Lovász ϑ function, well studied in graph theory and SVM formulation. This link allows us to design scalable algorithms for computationally difficult problems. In particular we show that on random graphs, the Lovász ϑ function can be well approximated by solving an SVM. Furthermore this property is not destroyed even when one plants a large clique in such graphs, allowing extremely scalable algorithms for identifying it. Using tools from MKL we further describe a algorithm for finding a common large dense region in large graphs. This algorithm achieves an order of magnitude scalability compared to state-of-the-art method based on exhaustive search of frequent quasi-cliques.

Acknowledgments

We would like to thank the anonymous reviewers for their useful comments. The authors also thank Fredrik Johansson and Christos Dimitrikakis for critically reading the manuscript.

Appendix A. Extended Results on DIMACS Graph Families

Figure 5 shows the densities of the induced subgraph $\gamma(G_{S_c}^{(l)})$ relative to original graph density $\gamma(G^{(l)})$ for all graphs $G^{(l)} \in \mathbb{G}$ at different values of $c \in [0,1]$ (i.e., different subgraph sizes $|S_c|$) for remaining graphs (other than those presented in Figure 2) in different DIMACS graph families.

Appendix B. Some Results Related to $\bar{G}(n,1-p,k)$

In this section we derive two results related to the planted clique problem. In particular we derive conditions when $\bar{G}(n,1-p,k)$ is a $\mathcal{Q}$ graph and we compute the ϑ function for the same graph.

B.1 When Is $\bar{G}(n,1-p,k)$ a $\mathcal{Q}$ Graph?

To prove Theorem 21 we will need a lower bound on the minimum eigenvalue of $\bar{G}$. One can prove the following

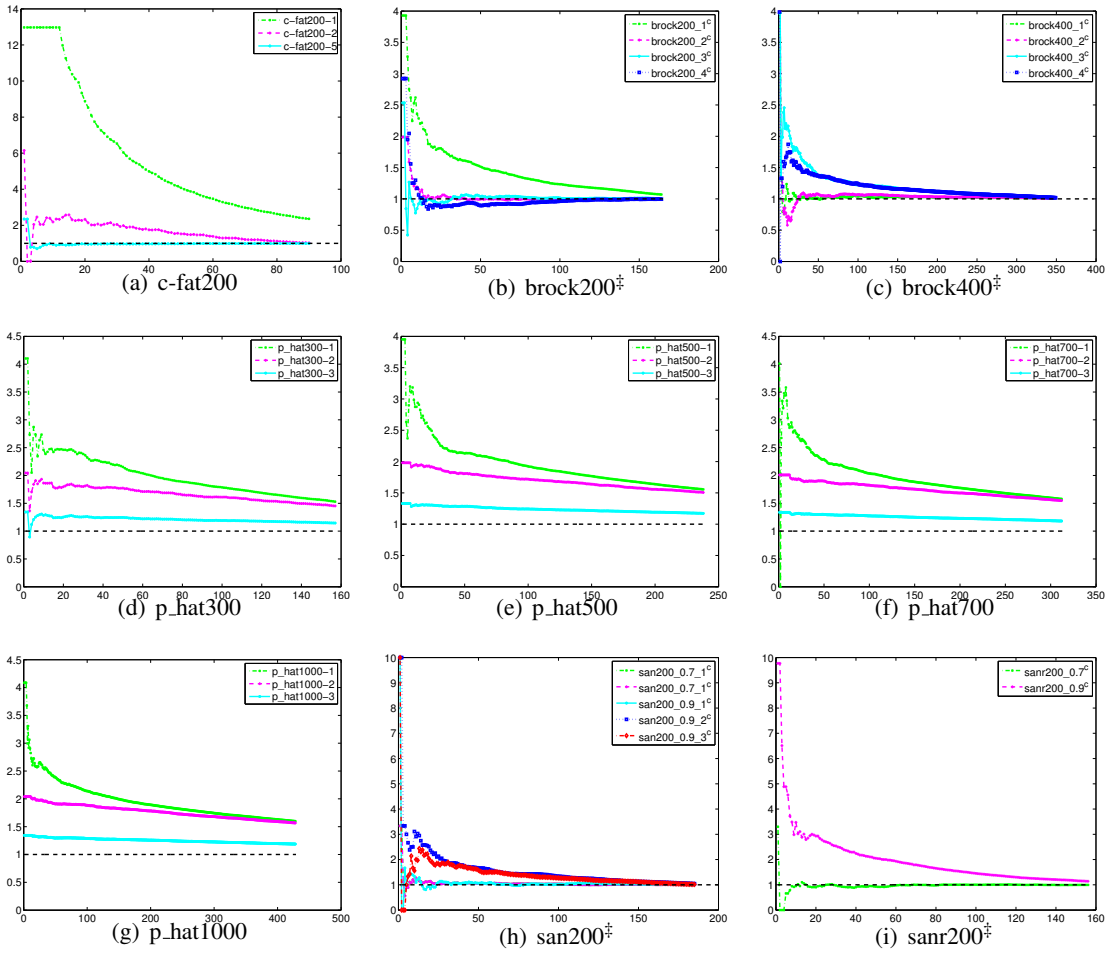


Figure 5: Common dense subgraph recovery. Figures $(a)-(i)$ show the densities of the induced subgraph $\gamma(G^{(l)}_{S_c})$ relative to original graph density $\gamma(G^{(l)})$ for all graphs $G^{(l)} \in \mathbb{G}$ at different values of $c \in [0,1]$ (i.e., different subgraph sizes $|S_c|$) for remaining graphs in different DIMACS graph families.

Lemma 31 *Let* $\mathbf{A}$ *be the adjacency matrix of* $\bar{G}(n, 1-p, k)$. *Then with probability at least* $1-\frac{1}{n}$, *the following holds:*

$$-\lambda_{\min}(\mathbf{A}) \leq kp - \frac{9k^2p}{16n} + \frac{\|\mathbf{A}-\mathbb{E}\mathbf{A}\|^2}{kp} + \sqrt{\ln n} + p.$$

The proof is crucially dependent on the following result.

Lemma 32 *Let* $\mathbf{A}$ *be the adjacency matrix of* $\bar{G}(n, 1-p, k)$. *For any fix* $\mathbf{x}$ *and any* $\delta > 0$ *we have*

$$Pr\left[\mathbf{x}^\top(\mathbf{A}-\mathbb{E}\mathbf{A})\mathbf{x} \leq -\delta\|\mathbf{x}\|^2\right] \leq e^{-\delta^2}.$$

Proof Let $\hat{\mathbf{A}} = \mathbf{A} - \mathbb{E}\mathbf{A}$ and consider the random variable

$$X = -\mathbf{x}^\top \hat{\mathbf{A}}\mathbf{x} = \sum_{i>j} -2x_i x_j \hat{A}_{ij},$$

where we note that the sum goes over independent random variables. Since $\hat{A}_{ij}$ varies by at most by 1, the term $2x_i x_j \hat{A}_{ij}$ varies at most $|2x_i x_j|$ where

$$\sum_{i>j} (2x_i x_j)^2 = 2\sum_{i\neq j} x_i^2 x_j^2 \leq 2\|\mathbf{x}\|^4.$$

Thus Hoeffding's inequality (McDiarmid, 1998) applied on X states that for any $t > 0$

$$Pr[X \geq t] \leq \exp\left(-\frac{t^2}{\|\mathbf{x}\|^4}\right).$$

The statement is obtained by letting $t = \delta\|\mathbf{x}\|^2$. ■

Proof [Proof of Lemma 31] Consider the matrix $\mathbf{A}' = \mathbf{A} + \mathbf{D}$ where $D_{ij} = p$ if $i = j \notin S$ and 0 otherwise, that is, adding p to all diagonal elements not in S. We note that this perturbation gives $\mathbb{E}\mathbf{A}'$ a 2-dimensional column space as all vectors in the column space are constant on S and $\bar{S}$ respectively. Let $\mathbf{x}$ be any unit vector in the column space where we write $\frac{\sin t}{\sqrt{k}}$ on S and $\frac{\cos t}{\sqrt{n-k}}$ on $\bar{S}$. Since all eigenvectors corresponding to non-zero eigenvalues must be in this vector space, the minimum eigenvalue of $\mathbb{E}\mathbf{A}'$ is either 0 or the minimum of

$$\begin{aligned}
\mathbf{x}^\top \mathbb{E}[\mathbf{A}']\mathbf{x} &= 2\sqrt{k(n-k)}p\sin t\cos t + (n-k)p\cos^2 t \\
&= \frac{(n-k)p}{2} + \sqrt{k(n-k)}p\sin 2t - \frac{(n-k)p}{2}\cos 2t \\
&\geq \frac{(n-k)p}{2} - \frac{p}{2}\sqrt{4k(n-k)+(n-k)^2} \\
&\geq \frac{(n-k)p}{2} - \frac{p}{2}\left(n + \frac{2nk-3k^2}{2n}\right) = -kp + \frac{3k^2 p}{4n}
\end{aligned}$$

where the last inequality follows by Taylor expansion of the square root at n^2. Since the last expression is non-positive we conclude that $\bar{\lambda} := -kp + \frac{3k^2 p}{4n}$ is a lower bound on the eigenvalues of $\mathbb{E}\mathbf{A}'$.

Now, let $\mathbf{v}$ be a normalized eigenvector corresponding to the minimum eigenvalue of $\mathbb{E}\mathbf{A}'$ and let $\mathbf{u} = \mathbf{x} + \mathbf{y}$ be any unit vector, where $\mathbf{x} \parallel \mathbf{v}$ and $\mathbf{y} \perp \mathbf{v}$. Since $\mathbb{E}\mathbf{A}'$ is a rank 2 matrix and clearly $\mathrm{Tr}(\mathbb{E}\mathbf{A}') \geq 0$, $\mathbb{E}\mathbf{A}'$ can at most have one negative eigenvalue and thus $\mathbf{y}^\top \mathbb{E}[\mathbf{A}']\mathbf{y} \geq 0$. Furthermore, Lemma 32 states that $\mathbf{v}^\top \hat{\mathbf{A}}\mathbf{v} \geq -\sqrt{\ln n}$ with probability at least $\left(1 - \frac{1}{n}\right)$. Thus,

$$\begin{aligned}
\lambda_{\min}(\mathbf{A}) &= \min_{\|\mathbf{u}\|=1} \mathbf{u}^\top \left(\mathbb{E}\mathbf{A}' + \hat{\mathbf{A}} - \mathbf{D}\right)\mathbf{u} \\
&\geq \min_{\|\mathbf{u}\|=1} \bar{\lambda}\|\mathbf{x}\|^2 + \mathbf{x}^\top \hat{\mathbf{A}}\mathbf{x} + (2\mathbf{x}+\mathbf{y})^\top \hat{\mathbf{A}}\mathbf{y} - \mathbf{u}^\top \mathbf{D}\mathbf{u} \\
&\geq -\sqrt{\ln n} - p + \min_{x_1^2 + x_2^2 = 1} \left(\bar{\lambda}x_1^2 - \|\hat{\mathbf{A}}\|\sqrt{4x_1^2 + x_2^2}\, x_2\right),
\end{aligned}$$

where we again use $\hat{\mathbf{A}} = \mathbf{A} - \mathbb{E}\mathbf{A}$. By substituting $x_2 = \frac{2}{\sqrt{3}}\cos t$ in the last term we get

$$\begin{aligned}\bar{\lambda}(1-x_2^2) - \|\hat{\mathbf{A}}\|\sqrt{4-3x_2^2}\,x_2 &= \frac{\bar{\lambda}}{3} - \frac{2}{3}\left(\bar{\lambda}\cos 2t + \sqrt{3}\|\hat{\mathbf{A}}\|\sin 2t\right) \\ &\geq \frac{\bar{\lambda}}{3} - \frac{2}{3}\sqrt{\bar{\lambda}^2 + 3\|\hat{\mathbf{A}}\|^2}\end{aligned}$$

so by Taylor expanding the square root at k^2p^2 we conclude that

$$\begin{aligned}\min_{x_1^2+x_2^2=1}\left(\bar{\lambda}x_1^2 - \|\hat{\mathbf{A}}\|\sqrt{4x_1^2+x_2^2}\,x_2\right) &\geq \frac{\bar{\lambda}}{3} - \frac{2}{3}\left(kp + \frac{\bar{\lambda}^2 + 3\|\hat{\mathbf{A}}\|^2 - k^2p^2}{2kp}\right) \\ &= -kp + \frac{k^2p}{n}\left(\frac{3}{4} - \frac{3k}{16n}\right) - \frac{\|\hat{\mathbf{A}}\|}{kp}.\end{aligned}$$

Thus with probability at least $\left(1-\frac{1}{n}\right)$, we get

$$\lambda_{\min}(\mathbf{A}) \geq -\sqrt{\ln n} - p - kp + \frac{k^2p}{n}\left(\frac{3}{4} - \frac{3k}{16n}\right) - \frac{\|\hat{\mathbf{A}}\|}{kp}.$$

■

Before we prove the main theorem we will need one more ingredient, a bound on the norm of the difference between the adjacency matrix and its expectation.

Lemma 33 *If* $\mathbf{A}$ *is the adjacency matrix of* $\bar{G}(n, 1-p, k)$ *and* $p(1-p) = \Omega(n^{-1}\log^4 n)$*, then almost surely* $\|\mathbf{A} - \mathbb{E}\mathbf{A}\| = O(\sqrt{np(1-p)})$.

Proof The Lemma is easily derived from Theorem 34, due to Vu (2007). In particular in the given regime of p, one can choose $\mathbf{R} = \mathbf{A} - \mathbb{E}\mathbf{A}$, $\sigma^2 = p(1-p)$ and $K = 1$. For such a choice, one can show $C(K\sigma)^{1/2}n^{1/4}\ln n = O(\sigma\sqrt{n})$ and the lemma follows by direct application of following result in Theorem 34. ■

We next state a Theorem, without proof, due to Vu (2007).

Theorem 34 *(Vu, 2007) There are constants C and C′ such that the following holds. Let* R_{ij}, $1 \leq i \leq j \leq n$ *be independent random variables, with* $\mathbb{E}(R_{ij}) = 0, \mathbb{E}(R_{ij}^2) \leq \sigma^2, |R_{ij}| \leq K$ *and* $\sigma \geq C'n^{-1/2}K\ln^2 n$*. Then with probability tending to 1 as* $n \to \infty$

$$\|\mathbf{R}\| \leq 2\sigma\sqrt{n} + C(K\sigma)^{1/2}n^{1/4}\ln n.$$

We are now ready to begin the proof of the main result.

Proof [Proof of Theorem 21] According to Theorem 5, G is a Q graph if

$$-\lambda_{\min}(\mathbf{A}) \leq |N(i) \cap S| \text{ for all } i \notin S, \tag{31}$$

where S denotes a maximum independent set. We will prove that this equation is satisfied almost surely for S denoting the planted independent set. In the specified parameter regime, one can argue that the planted set is the maximum independent set almost surely. It should however be noted

that this is strictly speaking not needed. As (31) is essentially reformulated optimality criteria, any independent set satisfying the equation must be a maximum independent set.

For each $i \notin S$, $|N(i) \cap S|$ has $Bin(k, p)$ distribution, so using Chernoff bound (Frieze and Reed, 1998) together with a union bound one obtains $|N(i) \cap S| \geq kp - \sqrt{6kp \ln n}$ for all $i \notin S$ with probability at least $1 - \frac{1}{n}$. Using this together with Lemma 31, we get that G is a Q graph with probability $1 - O(\frac{1}{n})$ if

$$kp - \frac{9k^2}{16n} + \frac{\|\mathbf{A} - \mathbb{E}\mathbf{A}\|^2}{kp} + \sqrt{\ln n} + p \leq kp - \sqrt{6kp \ln n}.$$

Application of Lemma 33 yields $\|\mathbf{A} - \mathbb{E}\mathbf{A}\|^2 = O(np(1-p))$. Clearly then almost surely G is a Q graph if the following holds,

$$\frac{9k^2 p}{16n} \geq \sqrt{6kp \ln n} + \sqrt{\ln n} + O\left(\frac{n}{k}\right). \tag{32}$$

Next we note that for $k \geq \frac{8}{3} n^{2/3} p^{-1/3} (\ln n)^{1/3}$ the following holds

$$\begin{aligned}
\sqrt{6kp \ln n} + \sqrt{\ln n} + O\left(\frac{n}{k}\right) &= \sqrt{6kp \ln n}\left(1 + O\left(\frac{1}{k^{1/2} p^{1/2}} + \frac{n}{k^{3/2} p^{1/2}}\right)\right) \\
&\leq \sqrt{6kp \ln n}\left(1 + O(\frac{1}{n^{1/3} p^{1/3} (\ln n)^{1/6}} + \frac{1}{(\ln n)^{1/2}})\right) \\
&= \sqrt{6kp \ln n}\,(1 + o(1))
\end{aligned}$$

where $np = \Omega(\log^4 n)$ by assumption in the theorem. This means that equation (32) is satisfied for

$$k \geq (1 + o(1)) \frac{8}{3} n^{2/3} p^{-1/3} (\ln n)^{1/3}$$

and the theorem follows. ■

B.2 Computation of ϑ Function for $\bar{G}(n, 1-p, k)$

Proof [of Theorem 22] We will base the proof on the following definition of ϑ function (Knuth, 1994),

$$\vartheta(G) = \min_{\mathbf{M}} \lambda_1(\mathbf{M})$$

where $\mathbf{M}$ goes over all symmetric $n \times n$ matrices such that $M_{ij} = 1$ whenever $A_{ij} = 0$, where $\mathbf{A}$ denotes the adjacency matrix of G. For the given graph $\bar{G}$ we will construct a qualified guess of $\mathbf{M}$ at optimality and use this to show a tight bound on $\vartheta(\bar{G})$. Specifically, let $\mathbf{A}$ be the adjacency matrix of $\bar{G}$ and consider the matrix $\mathbf{M}$ defined by

$$M_{ij} = \begin{cases} 1 & \text{if } 1 \leq i, j \leq k \\ 1 - r_j A_{ij} & \text{if } 1 \leq i \leq k < j \leq n \\ 1 - r_i A_{ij} & \text{if } 1 \leq j \leq k < i \leq n \\ \frac{1}{p}(p - A_{ij}) & \text{if } k < i, j \leq n \end{cases} \tag{33}$$

where r_i is chosen such that $\sum_{j=1}^{k} M_{ij} = 0$ for all $i > k$. Equivalently, r_i satisfies

$$k - S_i r_i = 0 \tag{34}$$

where S_i is the number of neighbours of i which shares an edge with the planted independent set. The case $S_i = 0$ for some $i > k$ may be resolved arbitrarily. We note that $S_i \sim Bin(k, p)$.

Let $\mathbf{e}_k$ denote the n-dimensional vector where the first k elements are 1 and the rest 0. Note that we constructed $\mathbf{M}$ such that it has $\mathbf{e}_k$ as an eigenvector with corresponding eigenvalue k. Using Lemma 35 we see that this is the maximum eigenvalue almost surely. To conclude we note that since there is an independent set of size k in $\bar{G}$, $k \leq \alpha(\bar{G}) \leq \vartheta(\bar{G})$. As noted above, $\vartheta(\bar{G}) \leq \lambda_1(\mathbf{M}) = k$ almost surely, so $k \leq \vartheta(\bar{G}) \leq k$, and thus $\vartheta(\bar{G}) = k$ almost surely. ■

Lemma 35 *Let* $\mathbf{M}$ *be defined as in above. If* $k > 2\sqrt{\frac{n(1-p)}{p}}\,(1+o(1))$ *and* $p(1-p) = \Omega(n^{-1}\log^4 n)$, *then* $\lambda_2(\mathbf{M}) < k$ *almost surely.*

Let $\mathbf{A}$, as before, denote the adjacency matrix of $\bar{G}$. Consider the two matrices,

$$U_{ij} = \begin{cases} 0 & \text{if } 1 \leq i, j \leq k \\ \frac{1}{p}(p - A_{ij}) & \text{if } 1 \leq i \leq k < j \leq n \\ \frac{1}{p}(p - A_{ij}) & \text{if } 1 \leq j \leq k < i \leq n \\ \frac{1}{p}(p - A_{ij}) & \text{if } k < i, j \leq n, \end{cases} \tag{35}$$

$$V_{ij} = \begin{cases} 0 & \text{if } 1 \leq i, j \leq k \\ \left(r_j - \frac{1}{p}\right)(p - A_{ij}) & \text{if } 1 \leq i \leq k < j \leq n \\ \left(r_i - \frac{1}{p}\right)(p - A_{ij}) & \text{if } 1 \leq j \leq k < i \leq n \\ 0 & \text{if } k < i, j \leq n \end{cases} \tag{36}$$

where r_i are the same as in (33).

The following corollary is a direct consequence of Theorem 34

Corollary 36 *For any symmetric* $n \times n$ *matrix* $\mathbf{R}$ *with entries* R_{ij}, *independent with mean* 0 *and variance bounded above by* $\frac{1-p}{p}$ *and* $|R_{ij}| \leq \frac{1}{p}$ *the following holds*

$$\|\mathbf{R}\| \leq 2\sqrt{\frac{n(1-p)}{p}}\,(1+o(1))$$

whenever $p(1-p) = \Omega(n^{-1}\log^4 n)$.

Proof The corollary follows by noticing that the matrix $\mathbf{R}$ satisfies the conditions of Theorem 34 with parameters $\sigma^2 = \frac{1-p}{p}$ and $K = \frac{1}{p}$. ■

We will use the corollary to obtain bounds on the largest eigenvalue of $\mathbf{U}$ and $\mathbf{V}$.

Lemma 37 *Let* $\mathbf{U}$ *be defined in* (35), *then if* $p(1-p) = \Omega(n^{-1}\log^4 n)$

$$\lambda_1(\mathbf{U}) \leq 2\sqrt{\frac{n(1-p)}{p}}\,(1+o(1))\,.$$

Proof Let $\mathbf{D}$ be diagonal matrix with $D_{ii} = 0$ whenever $1 \leq i \leq k$ and $D_{ii} = 1$ whenever $i = k+1, \ldots, n$. The matrix $\mathbf{U} - \mathbf{D}$ satisfies the condition of Corollary 36 and hence

$$\lambda_1(\mathbf{U}) \leq \|\mathbf{U} - \mathbf{D}\| + \|\mathbf{D}\| \leq 2\sqrt{\frac{n(1-p)}{p}}(1 + o(1)) + 1$$

where we have used the fact that $\|\mathbf{D}\| = 1$. ■

Lemma 38 *Let* $\mathbf{V}$ *be defined in* (36)*, then if* $p(1-p) = \Omega(n^{-1}\log^4 n)$ *and* $k > 2\sqrt{\frac{n(1-p)}{p}}$

$$\lambda_1(\mathbf{V}) \leq o\left(\sqrt{\frac{n(1-p)}{p}}\right).$$

Proof Consider the $n \times n$ matrix $\mathbf{V}'$ defined by

$$V'_{ij} = \begin{cases} 0 & \text{if } 1 \leq i, j \leq k \\ \frac{1}{p}(p - A_{ij}) & \text{if } 1 \leq i \leq k < j \leq n \\ \frac{1}{p}(p - A_{ij}) & \text{if } 1 \leq j \leq k < i \leq n \\ 0 & \text{if } k < i, j \leq n. \end{cases}$$

We can apply Corollary 36 to obtain $\|\mathbf{V}'\| = O\sqrt{\frac{n(1-p)}{p}}$.

Let $\mathbf{x}$ be a unit eigenvector corresponding to the maximum eigenvalue of $\mathbf{V}$. This means that

$$\begin{aligned} \lambda_1(\mathbf{V}) &= \mathbf{x}^\top \mathbf{V}\mathbf{x} \\ &= 2\sum_{i=k+1}^{n} x_i \left(r_i - \frac{1}{p}\right) \sum_{j=1}^{k} (p - A_{ij}) x_j \end{aligned}$$

and by Cauchy-Schwartz inequality

$$\begin{aligned} &\leq 2\left(\sum_{i=k+1}^{n} x_i^2 \cdot \sum_{i=k+1}^{n} \left(r_i - \frac{1}{p}\right)^2 \left(\sum_{j=1}^{k} (p - A_{ij}) x_j\right)^2\right)^{1/2} \\ &\leq 2\left(\sum_{i=k+1}^{n} (pr_i - 1)^2 \left(\sum_{j=1}^{k} \frac{1}{p}(p - A_{ij}) x_j\right)^2\right)^{1/2} \\ &= 2\left(\sum_{i=k+1}^{n} (pr_i - 1)^2 \left(\sum_{j=1}^{k} V'_{ij} x_j\right)^2\right)^{1/2} \leq 2\max_i |pr_i - 1| \cdot \left(\sum_{i=k+1}^{n} \left(\sum_{j=1}^{k} V'_{ij} x_j\right)^2\right)^{1/2} \end{aligned}$$

since $V'_{ij} = 0$ for $i, j > k$

$$
\begin{aligned}
&= 2\max_i |pr_i - 1| \cdot \left(\sum_{i=k+1}^{n} \left(\sum_{j=1}^{n} V'_{ij} x_j \right)^2 \right)^{1/2} \\
&\leq 2\max_i |pr_i - 1| \cdot \left(\sum_{i=1}^{n} \left(\sum_{j=1}^{n} V'_{ij} x_j \right)^2 \right)^{1/2} \\
&= 2\max_i |pr_i - 1| \cdot \|\mathbf{V}'\mathbf{x}\| = \max_i |pr_i - 1| \cdot O\left(\sqrt{\frac{n(1-p)}{p}} \right),
\end{aligned}
$$

where the last step follows from that $\|\mathbf{V}'\mathbf{x}\| \leq \|\mathbf{V}'\| \cdot \|\mathbf{x}\| = \|\mathbf{V}'\|$.

By the definition of $\mathbf{M}$, see (34), $r_i = \frac{k}{S_i}$, where $S_i \sim Bin(k, p)$. For such random variables, we have the following Chernoff bound

Lemma 39 (McDiarmid, 1998) *For every $0 < a \leq kp$ we have*

$$Pr\left[|S_i - kp| \geq a\right] \leq 2e^{-a^2/3kp}.$$

Let $a = \sqrt{6kp \ln n}$. Using the assumptions in the lemma it is easy to verify that $kp > 2\sqrt{np(1-p)} \gg \ln^2 n \gg \ln n$. Thus for large enough n we have $a \leq kp$, so in that case we know that $|S_i - kp| \geq a$ for some $i > k$ happens with at most probability $n \cdot e^{-2\ln n} = o(1)$. This means that, almost surely

$$\max_i |pr_i - 1| = |\frac{kp}{kp + O\sqrt{kp \ln n}} - 1| = \frac{O\sqrt{kp \ln n}}{kp + O\sqrt{kp \ln n}} = O\sqrt{\frac{\ln n}{kp}} = o(1),$$

where we use that $\ln n \ll kp$ as noted above. Thus $\lambda_1(V) = o\sqrt{\frac{n(1-p)}{p}}$ almost surely. ■

We are now in a position to begin the proof of Lemma 35.

Proof [Proof of Lemma 35] Let $\mathbf{e}_k$ denote the n-dimensional vector where the first k elements are 1 and the rest 0. By the variational inequality

$$
\begin{aligned}
\lambda_2(\mathbf{M}) &= \min_{\mathbf{v}} \max_{\|\mathbf{x}\|=1 \, \mathbf{x} \perp \mathbf{v}} \mathbf{x}^\top \mathbf{M} \mathbf{x} \\
&\leq \max_{\|\mathbf{x}\|=1 \, \mathbf{x} \perp \mathbf{e}_k} \mathbf{x}^\top \mathbf{M} \mathbf{x} \\
&\leq \lambda_1(\mathbf{U}) + \lambda_1(\mathbf{V}) + \max_{|\mathbf{x}|=1 \, \mathbf{x} \perp \mathbf{e}_k} \mathbf{x}^\top (\mathbf{M} - \mathbf{U} - \mathbf{V}) \mathbf{x}.
\end{aligned}
$$

Recalling the definitions of $\mathbf{M}$, $\mathbf{U}$ and $\mathbf{V}$, see (33), (35), (36), we note that for all $\mathbf{x} \perp \mathbf{e}_k$

$$\begin{aligned}
&\mathbf{x}^\top(\mathbf{M}-\mathbf{U}-\mathbf{V})\mathbf{x} \\
&= \sum_{i=1}^{k}\sum_{j=1}^{k} x_i(M_{ij}-U_{ij}-V_{ij})x_j + 2\sum_{i=k+1}^{n}\sum_{j=1}^{k} x_i(M_{ij}-U_{ij}-V_{ij})x_j \\
&\quad + \sum_{i=k+1}^{n}\sum_{j=k+1}^{n} x_i(M_{ij}-U_{ij}-V_{ij})x_j \\
&= \sum_{i=1}^{k}\sum_{j=1}^{k} x_i x_j + 2\sum_{i=k+1}^{n}\sum_{j=1}^{k} x_i\left(1 - r_i A_{ij} - \frac{1}{p}(p-A_{ij}) - \left(r_i - \frac{1}{p}\right)(p-A_{ij})\right)x_j \\
&= \sum_{i=1}^{k}\sum_{j=1}^{k} x_i x_j + 2\sum_{i=k+1}^{n}\sum_{j=1}^{k} x_i(1-pr_i)x_j \\
&= \sum_{i=1}^{k} x_i \underbrace{\left(\sum_{j=1}^{k} x_j\right)}_{=0} + 2\sum_{i=k+1}^{n} x_i(1-pr_i)\underbrace{\left(\sum_{j=1}^{k} x_j\right)}_{=0} = 0
\end{aligned}$$

so $\max_{\|\mathbf{x}\|=1,\,\mathbf{x}\perp\mathbf{e}_k} x^\top(\mathbf{M}-\mathbf{U}-\mathbf{V})\mathbf{x} = 0$. Thus if we assume that $k > 2\sqrt{\frac{n(1-p)}{p}}$, Lemmas 37 and 38 imply that

$$\lambda_2(\mathbf{M}) \le \lambda_1(\mathbf{U}) + \lambda_1(\mathbf{V}) \le 2\sqrt{\frac{n(1-p)}{p}}\,(1+o(1))\,.$$

The Lemma follows by letting k be strictly greater than the maximum of this value and $2\sqrt{\frac{n(1-p)}{p}}$. ■

References

J. Aflalo, A. Ben-Tal, C. Bhattacharyya, J.S. Nath, and S. Raman. Variable sparsity kernel learning. *Journal of Machine Learning Research*, 12:565–592, 2011.

N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. *Random Structures and Algorithms*, pages 457–466, 1998.

B. Applebaum, B. Barak, and A. Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC '10)*, pages 171–180, 2010.

S. Arora, B. Barak, M. Brunnermeier, and R. Ge. Computational complexity and information asymmetry in financial products (extended abstract). In *First Symposium on Innovations in Computer Science*, ICS, pages 49–65, 2010.

C. Bachoc, D. Gijswijt, A. Schrijver, and F. Vallentin. Invariant semidefinite programs. In M. F. Anjos and J. B. Lasserre, editors, *Handbook on Semidefinite, Conic and Polynomial Optimization*, volume 166 of *International Series in Operations Research & Management Science*, pages 219–269. Springer US, 2012.

B. Bollobás. *Modern Graph Theory*. Springer Verlag, 1998.

B. Bollobás. *Random Graphs*. Cambridge University Press, 2001.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

A. E. Brouwer and W. H. Haemers. *Spectra of Graphs*. Springer Verlag, 2012.

T.-H. H. Chan, K. L. Chang, and R. Raman. An sdp primal-dual algorithm for approximating the lovász-theta function. In *IEEE International Symposium on Information Theory*, ISIT '09, pages 2808–2812, 2009.

A. Coja-Oghlan. The lovász number of random graphs. *Combinatorics, Probability & Computing*, 14(4):439–465, 2005.

A. Coja-Oghlan and A. Taraz. Exact and approximative algorithms for coloring g(n, p). *Random Structures and Algorithms*, 24(3):259–278, 2004.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.

L. Devroye, A. György, G. Lugosi, and F. Udina. High-dimensional random geometric graphs and their clique number. *Electronic Journal of Probability*, 16, 2011.

D. P. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

U. Feige and R. Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures and Algorithms*, 16:195–208, 2000.

A. M. Frieze and C. McDiarmid. Algorithmic theory of random graphs. *Random Structures and Algorithms*, 10(1-2):5–42, 1997.

A. M. Frieze and B. Reed. Probabilistic analysis of algorithms. In M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*, volume 16 of *Algorithms and Combinatorics*, pages 36–92. Springer-Verlag, Berlin, 1998.

Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1:233–241, 1981.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

J. Håstad. Clique is hard to approximate within $n^{1-\varepsilon}$. *Acta Mathematica*, 182(1):105–142, 1999.

A. J. Hoffman and L. Howes. On eigenvalues and colorings of graphs, ii. *Annals of the New York Academy of Sciences*, 175(1):238–242, 1970.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

E. Hu, B. Wang, and S. Chen. Bcdnpkl: Scalable non-parametric kernel learning using block coordinate descent. In *Proceedings of the 28th International Conference on Machine Learning*, ICML '11, pages 209–216, 2011.

H. Hu, X. Yan, Y. Huang, J. Han, and X. J. Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, 21(suppl 1):i213–i221, 2005.

D. R. Hush, P. Kelly, C. Scovel, and I. Steinwart. Qp algorithms with guaranteed accuracy and run time for support vector machines. *Journal of Machine Learning Research*, 7:733–769, 2006.

S. Janson, T. Luczak, and A. Rucinski. *Random Graphs*. John Wiley & Sons, Inc., 2000.

M. Jerrum. Large cliques elude the metropolis process. *Random Structures and Algorithms*, 3(4): 347–360, 1992.

D. Jiang and J. Pei. Mining frequent cross-graph quasi-cliques. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(4):16, 2009.

D. S. Johnson and M. A. Trick. *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, October 11-13, 1993*, volume 26. Amer Mathematical Society, 1996.

F. Juhász. The asymptotic behaviour of lovász' theta-function for random graphs. *Combinatorica*, 2(2):153–155, 1982.

D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45:246–265, March 1998. ISSN 0004-5411.

D. Knuth. The sandwich theorem. *Electronic Journal of Combinatorics*, 1(A1), 1994.

M. Krivelevich. Deciding k-colorability in expected polynomial time. *Information Processing Letters*, 81, pages 1–6, 2002.

L. Kucera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57 (2-3):193–212, 1995.

G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

K. J. Lang and R. Andersen. Finding dense and isolated submarkets in a sponsored search spending graph. In *Proceedings of the ACM Sixteenth Conference on Information and Knowledge Management (CIKM '07)*, pages 613–622, 2007.

V. E. Lee, N. Ruan, R. Jin, and C. Aggarwal. A survey of algorithms for dense subgraph discovery. *Managing and Mining Graph Data*, pages 303–336, 2010.

L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25(1): 1–7, 1979.

C. J. Luz. An upper bound on the independence number of a graph computable in polynomial-time. *Operations Research Letters*, 18(3):139 – 145, 1995.

C. J. Luz and A. Schrijver. A convex quadratic characterization of the lovász theta number. *SIAM Journal on Discrete Mathematics*, 19(2):382–387, 2006.

C. McDiarmid. Concentration. In M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*, volume 16 of *Algorithms and Combinatorics*, pages 1–46. Springer-Verlag, Berlin, 1998.

M. E. J. Newman, A. L. Barabasi, and D. J. Watts. *The Structure and Dynamics of Networks*. Princeton Univ Pr, 2006.

P. Pardalos and S. Rebennack. Computational challenges with cliques, quasi-cliques and clique partitions in graphs. *Experimental Algorithms*, pages 13–22, 2010.

Y. Podolyan and G. Karypis. Common pharmacophore identification using frequent clique detection algorithm. *Journal of Chemical Information and Modeling*, 49(1):13–21, 2009.

A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

B. Schölkopf, J. C. 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.

M. Sion. On general minimax theorems. *Pac. J. Math.*, 8:171–176, 1958.

V. Spirin and L. A. Mirny. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences*, 100(21):12123, 2003.

Y. Takahashi, Y. Satoh, H. Suzuki, and S. Sasaki. Recognition of largest common structural fragment among a variety of chemical structures. *Analytical Sciences*, 3(1):23–28, 1987.

M. Talagrand. Concentration of measure and isoperimetric inequalities in product spaces. *Publications Mathematiques de L'IHES*, 81:73–205, 1995.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

V. H. Vu. Spectral norm of random matrices. *Combinatorica*, 27(6):721–736, 2007.

Journal of Machine Learning Research 14 (2013) 3537-3559 Submitted 2/13; Revised 8/13; Published 12/13

Learning Trees from Strings: A Strong Learning Algorithm for some Context-Free Grammars

Alexander Clark ALEXANDER.CLARK@KCL.AC.UK
Department of Philosophy
King's College London
The Strand
London WC2R 2LS

Editor: Mehryar Mohri

Abstract

Standard models of language learning are concerned with weak learning: the learner, receiving as input only information about the strings in the language, must learn to generalise and to generate the correct, potentially infinite, set of strings generated by some target grammar. Here we define the corresponding notion of strong learning: the learner, again only receiving strings as input, must learn a grammar that generates the correct set of structures or parse trees. We formalise this using a modification of Gold's identification in the limit model, requiring convergence to a grammar that is isomorphic to the target grammar. We take as our starting point a simple learning algorithm for substitutable context-free languages, based on principles of distributional learning, and modify it so that it will converge to a canonical grammar for each language. We prove a corresponding strong learning result for a subclass of context-free grammars.

Keywords: context-free grammars, grammatical inference, identification in the limit, structure learning

1. Introduction

We present an algorithm for inducing a context-free grammar from a set of strings; this algorithm comes with a strong theoretical guarantee: it works in polynomial time, and for any grammar in a certain class it will converge to a grammar which is isomorphic/strongly equivalent to the target grammar. Moreover the convergence is rapid in a technical sense. This very strong guarantee comes of course at a price: the class of grammars is small. In the first part of the paper we explain the learning model we use which is an extension of the Gold identification in the limit model; and in the second part we present an algorithm which learns a class of languages with respect to this model. We have implemented this algorithm and we present some examples at the end which illustrate the properties of this algorithm, testing on some simple example languages. As far as we are aware this is the first nontrivial algorithm for learning trees from strings which has any sort of theoretical guarantee of its convergence and correctness.

Our ultimate domain of application of these techniques is primarily in linguistics, where the strings will be sequences of words in a natural language, but the techniques can be applied more broadly to artificial languages, bioinformatics and other fields where the input data consists of strings which have some hierarchical structure.

We can contrast the approach here with the task of unsupervised parsing in computational linguistics as exemplified by Cohn et al. (2010). Unsupervised parsers use a variety of heuristic approaches to extract a single tree for each sentence, taking as input a large natural language corpus, and being evaluated against some linguistically annotated corpus. Here we are interested not in finding the most likely parse, but in finding the set of allowable parses in a theoretically well-founded way.

1.1 Linguistics

The notions of weak and strong generation are fundamental in the fields of mathematical and theoretical linguistics. A formal grammar weakly generates a set of strings, and strongly generates a set of structures (Miller, 1999). We do not have the space for a full discussion of the rather subtle methodological and indeed philosophical issues involved with which model is appropriate for studying linguistics, which questions depend on what the subject matter of linguistics is taken to be; we merely note that while mathematical attention has largely focused on the issues of weak generation, many linguists are more concerned with the issues of strong generation and as a result take the weak results to be largely irrelevant (Berwick et al., 2011). Indeed, taking a grammar as a model of human linguistic competence, we are primarily interested in the set of structures generated. Unfortunately, we have little or no direct evidence about the nature of these structures, notwithstanding recent advances in neuroimaging and psycholinguistics, and our sources of information are essentially only about the set of strings that are weakly generated by the grammar, since these can be observed, and our intuitions about the associated meanings.

We can define corresponding notions of weak and strong learning.[1] Weak learning involves merely learning a grammar that generates the right set of strings; strong learning involves learning a grammar that generates the right set of structures (Wexler and Culicover, 1980, p. 58). Some sentences are ambiguous and will require a grammar that generates more than one structure for a particular sentence. We do not consider in this paper the problem of learning when the input to the learner are *trees*; see for example Sakakibara (1990, 1992), Drewes and Högberg (2003) and López et al. (2004). We consider only the problem where the learner has access to the flat strings alone, but must infer an appropriate set of trees for each string in the language. Rather than observing the derivation trees themselves, we observe only the yields of the trees.

Weak learning of context-free grammars and richer formalisms has made significant progress in recent years (Clark and Eyraud, 2007; Yoshinaka, 2011; Yoshinaka and Kanazawa, 2011; Yoshinaka, 2012) but strong learning has received less attention. For CFGs this means that the hypothesis needs to be isomorphic (assuming it is trim) or better yet, *identical* to the target grammar. We define these notions of equivalence and the associated learning models in Section 3. Strong learning is obviously impossible for the full class of context-free grammars since there are an infinite number of structurally different context-free grammars that generate a given context-free language.

In this paper we work in a categorical model which assumes, unrealistically, a partition of the strings into grammatical and ungrammatical, but probabilistically the situation is not better; given a distribution defined by a probabilistic CFG (PCFG) there are infinitely many structurally different CFGs that define the same set of distributions; in other words PCFGs are not identifiable from strings (Hsu et al., 2013). This is in contrast to discrete HMMs which are (Petrie, 1969).

1. Note that this has nothing to do with strong and weak learners as those terms are used in the boosting literature in machine learning (Schapire, 1999).

The contributions of this paper are as follows. We first define an appropriate notion of strong learning from strings, restricting ourselves to the case of CFGs for simplicity. We then show that existing learning algorithms for regular languages (Angluin, 1982) can be viewed as also being strong learning algorithms, in a trivial sense. We then present a strong learning algorithm for some CFGs, based on combining the polynomial algorithm for substitutable context-free languages defined in Clark and Eyraud (2007), which we recall in Section 4, with a recent proposal for a formal notion of syntactic structure (Clark, 2011) that we interpret as a form of canonical grammars. We specify the canonical grammars we target in Section 5, present an algorithm in Section 6, and prove its correctness and efficiency in Section 7. Section 8 contains some examples, including one with an ambiguous grammar. An appendix contains some detailed proofs of various technical lemmas regarding the properties of the languages we consider in this paper.

2. Notation

Let Σ be a finite non-empty set of atomic symbols. Σ^* is the set of all finite strings over Σ. We denote the empty string by λ. The set of non-empty strings is $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$. We write $|u|$ for the length of a string u, and for a finite set of strings X we define the size as $\|X\| = \sum_{w \in X} |w|$.

A language L is any subset of Σ^*. Given two languages $M, N \subseteq \Sigma^*$ we write $M \cdot N$ or sometimes just MN for the set $\{uv \mid u \in M, v \in N\}$. Note that this is just the normal concatenation of sets of strings.

Given a language D, we define $\mathrm{Sub}(D)$ to be the set of non-empty substrings of elements of D:

$$\mathrm{Sub}(D) = \{u \in \Sigma^+ \mid \exists (l,r) \in \Sigma^* \times \Sigma^*, lur \in D\}.$$

Given a non-zero sequence of languages $\alpha = \langle X_1, \ldots, X_n \rangle$ we write $\bar{\alpha}$ for the concatenation, that is, $\bar{\alpha} = X_1 \cdot \cdots \cdot X_n$. We shall assume an order $\prec$ or $\preceq$ on Σ which we shall extend to the length-lexicographic order on Σ^*.

We define a context (l,r) to be an ordered pair of strings, an element of $\Sigma^* \times \Sigma^*$. The distribution of a string $u \in \Sigma^*$ with respect to a language L is defined to be

$$D_L(u) = \{(l,r) \in \Sigma^* \times \Sigma^* \mid lur \in L\}.$$

We say that $u \equiv_L v$ iff $D_L(u) = D_L(v)$. This is the syntactic congruence, which is equivalent to complete mutual substitutability of u and v.

We write $[u]_L$ for $\{v \in \Sigma^* \mid D_L(u) = D_L(v)\}$. If we have a set of strings, X, that are all congruent then we write $[X]$ for the congruence class containing them. Note that for any strings u, v, $[uv] \supseteq [u][v]$ so if X, Y are congruence classes we can write $[XY]$ and the result is well defined.

The unique congruence class $[\lambda]$ is called the unit congruence class. The set $\{u \mid D_L(u) = \emptyset\}$ if it is non-empty is a congruence class, which is called the zero congruence class. A congruence class in a language L is non-zero iff it is a subset of $\mathrm{Sub}(L)$. We are mostly concerned with non-zero non-unit congruence classes in this paper.

Definition 1 *We will be considering sequences of congruence classes: so if α is a sequence $X_1, \ldots, X_n$ where each of the X_i is a congruence class, then we write $\bar{\alpha}$ for the set of strings formed by concatenating all of the X_i. We write $|\alpha|$ for the length of the sequence, n in this case. Note that all of the elements of $\bar{\alpha}$ will be congruent: if $u, v \in \bar{\alpha}$ then $u \equiv_L v$. We can therefore write without ambiguity $[\bar{\alpha}]$ for the congruence class of the strings in $\bar{\alpha}$.*

We say that $u \doteq_L v$ if there is some (l,r) such that $lur \in L$ and $lvr \in L$. This is partial or weak substitutability; u and v can be substituted for each other in the context (l,r). If $u \equiv_L v$ and u,v have a non-empty distribution then $u \doteq_L v$, but the converse is clearly not true.

Definition 2 *A language L is* substitutable *if for all $u,v \in \Sigma^+$, $u \doteq_L v$ implies $u \equiv_L v$.*

In other words, for any two non-empty strings u,v if $D_L(u) \cap D_L(v) \neq \emptyset$ then $D_L(u) = D_L(v)$. This language theoretic closure property allows us to define algorithms that generalise correctly, even under pessimistic learning conditions.

2.1 Context-Free Grammars

A context-free grammar (CFG) G is a tuple $G = \langle \Sigma, V, I, P \rangle$ where V is a finite non-empty set of nonterminals disjoint from Σ, $I \subseteq V$ is a set of distinguished start symbols and P is a subset of $V \times (V \cup \Sigma)^+$ called the set of productions. We write this as $N \to \alpha$. We do not allow productions with a right hand side of length 0, and as a result the languages we consider will not contain the empty string. We use $\mathcal{G}_{\text{CFG}}$ for the class of all context-free grammars.

We define the standard notion of single-step derivation as $\Rightarrow$ and define $\stackrel{*}{\Rightarrow}$ as the reflexive transitive closure of $\Rightarrow$; for all $N \in V$, $\mathcal{L}(G,N) = \{w \in \Sigma^* \mid N \stackrel{*}{\Rightarrow} w\}$; and $\mathcal{L}(G) = \bigcup_{S \in I} \mathcal{L}(G,S)$. Using a set of start symbols rather than a single start symbol does not change the generative capacity of the formalism.

We say that a CFG is trim if for every nonterminal N there is a context (l,r) such that $S \stackrel{*}{\Rightarrow} lNr$ for some $S \in I$ and a string u such that $N \stackrel{*}{\Rightarrow} u$: in other words every nonterminal can be used in the derivation of some string.

We say that two CFGs, G and G' are weakly equivalent if $\mathcal{L}(G) = \mathcal{L}(G')$.

Proposition 3 *(Ginsburg, 1966) Given two CFGs, G and G', it is undecidable whether $\mathcal{L}(G) = \mathcal{L}(G')$.*

Two CFGs are isomorphic if there is a bijection between the two sets of nonterminals which extends to a bijection between the productions. In other words they are identical up to a relabeling of nonterminals. We denote this by $G \cong G'$. Clearly if two grammars are isomorphic then they are weakly equivalent.

Proposition 4 *Given two CFGs, G and G', it is decidable whether $G \cong G'$.*

There is a trivial exponential time algorithm that involves searching through all possible bijections. This problem is GI-complete: as hard as the problem of graph isomorphism (Zemlyachenko et al., 1985; Read and Corneil, 1977). We may not be able to do this efficiently for general CFGs.

3. Learning Models

We start by reviewing the basic theory of learnability using the Gold identification in the limit paradigm (Gold, 1967). We consider only the model of given text—where the learner is provided with positive data only. We assume a class of CFGs, $\mathcal{G} \subseteq \mathcal{G}_{\text{CFG}}$.

A presentation of a language L is an infinite sequence of elements of Σ^*, $w_1, w_2, \ldots$ such that $L = \{w_i \mid i > 0\}$. Given a presentation $T = \langle w_1, \ldots \rangle$, we write T_n for the finite subsequence consisting

of the first n elements. A polynomial learning algorithm is a polynomially computable function from finite sequences of positive examples to $\mathcal{G}_{\text{CFG}}$.

Given a presentation T of some language L, we can apply A to the various prefixes of T, which produces an infinite sequence of elements of $\mathcal{G}_{\text{CFG}}$, $A(T_1), A(T_2), \ldots$. These are hypothesis grammars; we will use G_i to refer to $A(T_i)$, the ith hypothesis output by the learning algorithm.

Consider a target grammar $G_* \in \mathcal{G}$, and a sequence of hypothesized grammars $G_1, G_2, \ldots$ produced by a learning algorithm on a presentation T for $\mathcal{L}(G_*)$. There are various notions of convergence of which we outline four, which vary on two dimensions: one dimension concerns whether we are interested in weak or strong learning, and the other whether we are interested in controlling the number of internal changes as well, or are only interested in the external behaviour.

Weak behaviorally correct learning (WBC)
There is an N such that for all $n > N$, $\mathcal{L}(G_n) = \mathcal{L}(G_*)$.

Weak Gold learning (GOLD)
There is an N such that for all $n > N$, $\mathcal{L}(G_n) = \mathcal{L}(G_*)$ and $G_n = G_N$.

Strong behaviorally correct learning (SBC)
There is an N such that for all $n > N$, $G_n \cong G_*$.

Strong Gold learning (SGOLD)
There is an N such that for all $n > N$, $G_n \cong G_*$ and $G_n = G_N$.

For each of these four notions of convergence, we have a corresponding notion of learnability. We say that a learner, A, WBC/GOLD/SBC/SGOLD learns a grammar G_*, iff for every presentation of $\mathcal{L}(G_*)$, it WBC/GOLD/SBC/SGOLD converges on that presentation. Given a class of CFGs, $\mathcal{G}$, we say that A WBC/GOLD/SBC/SGOLD learns the class, iff for every G_* in $\mathcal{G}$ the learner WBC/GOLD/SBC/SGOLD learns G_*.

In the case of GOLD learning, this coincides precisely with the standard model of Gold identification in the limit from given text (positive data only) (Gold, 1967). WBC-learning is the standard model of behaviorally correct learning (Case and Lynes, 1982). We cannot in general turn a WBC-learner into a GOLD-learner: see discussion in Osherson et al. (1986). The property of *order-independence* as defined by Blum and Blum (1975), can be thought of as an even stronger version of SGOLD learning.

However, a SBC-learner can be changed into a SGOLD-learner, if we can test whether two hypotheses are isomorphic. There does not seem to be a theoretically interesting difference between SBC-learning and SGOLD-learning: the only difference, in the case of CFGs, is that the SBC learner may occasionally pick different labels for the nonterminals after convergence, whereas the SGOLD learner may not.

We can ask how can a GOLD learner differ from a SGOLD learner: how can a weak learner fail to be a strong learner? The difference is that on different presentations of the same language, a weak Gold learner may converge to different answers. That is to say we might have a learner which on presentation T' of grammar G produces a grammar G' and on presentation T'' of the same grammar, produces a grammar G'', where G' and G'' are weakly equivalent but not isomorphic.

Definition 5 *We say that a class of context-free grammars is redundant if it contains two grammars G_1, G_2 such that $G_1 \ncong G_2$ and $\mathcal{L}(G_1) = \mathcal{L}(G_2)$.*

Proposition 6 *Suppose that A is an algorithm which* SGOLD*-learns a class of grammars $\mathcal{G}$. Then $\mathcal{G}$ is not redundant.*

The proof is immediate—any presentation for G_1 is also a presentation for G_2. In other words if $\mathcal{G}$ contains two non-isomorphic grammars for the same language then it is not strongly learnable. A simple corollary is then that the class of CFGs is not strongly identifiable in the limit even from informed data, that is to say from labelled positive and negative examples, since there are an infinite number of non-isomorphic grammars for every non-empty language.

One can therefore try to convert a weak learner to a strong learner by defining a canonical form. If we can restrict the class so that there is only one structurally distinct grammar for each language, and we can compute that, then we could find a strong learning algorithm. We formalise this as follows. Suppose A is some learning algorithm that outputs grammars in a hypothesis class $\mathcal{H}_{\mathcal{A}} \subseteq \mathcal{G}_{\text{CFG}}$, and suppose that it can GOLD-learn the class of grammars $\mathcal{G} \subseteq \mathcal{H}_{\mathcal{A}}$. Suppose we have some 'canonicalisation' function f from $\mathcal{H}_{\mathcal{A}} \to \mathcal{G}_{\text{CFG}}$ such that for each $G \in \mathcal{G}$, $\mathcal{L}(G) = \mathcal{L}(f(G))$ and such that $f(\mathcal{G})$ is not redundant. Then we can construct a learner A' which outputs $A'(T_i) = f(A(T_i))$, which will then be a SGOLD learner for $\mathcal{G}$. Moreover, if A and f are both polynomially computable then so will A' be.

For example, suppose $\mathcal{D}$ is the class of all DFAs and f is the standard function for minimizing a deterministic finite-state automaton (DFA), which can be done in polynomial time. Since all minimal DFAs for a given regular language are isomorphic, $f(\mathcal{D})$ is not redundant. Therefore any learner for regular languages that outputs DFAs, such as the one in Angluin (1982), can be converted into a strong learner using this technique. From the point of view of structural learning such results are trivial in two important respects. The first is that each string in the language has exactly one labelled structure, and the other is that every structure is uniformly right branching, whereas we are interested in learning grammars which may assign more than one different structure to a given string.

Moreover, it is easy to see that any SGOLD-learner for a class of grammars $\mathcal{G}$ will implicitly define such a canonicalisation function for $\mathcal{G}$. We can enumerate the strings in the language and apply the learner to them, and the limit of the hypothesis grammars will then satisfy the conditions given above, though this function may not be computable. There is therefore a close relationship between canonicalisers and strong learners. There is much more that could be said about the learning models, and further refinements of them, but this is enough for our purposes.

4. Weak Learning of Substitutable Languages

We recall the Clark and Eyraud (2007) result, using a simplified version of the algorithm (Yoshinaka, 2008), and explain why it is only a weak rather than a strong result.

Given a finite non-empty set of strings $D = \{w_1, \ldots, w_n\}$ the learner constructs a grammar as shown in Algorithm 1. We create a set of symbols in bijection with the elements of $\text{Sub}(D)$ where we write $[[u]]$ for the symbol corresponding to the substring u: that is to say we have one nonterminal for each substring of the observed data. The grammar $\hat{G}(D)$ is the CFG $\langle \Sigma, V, I, P_L \cup P_B \cup P_U \rangle$ as shown in the pseudocode in Algorithm 1. The sets P_L, P_B and P_U are the sets of lexical, branching and unary productions respectively.

Example 1 *Given a set $D = \{c, acb\}$, we have $\text{Sub}(D) = \{a, b, c, ac, cb, acb\}$, and corresponding sets: $V = \{[[a]], [[b]], [[c]], [[ac]], [[cb]], [[acb]]\}$, $I = \{[[c]], [[acb]]\}$, $P_L = \{[[a]] \to a, [[b]] \to b, [[c]] \to$*

Algorithm 1 Grammar construction procedure

Data: A finite set of strings $D = \{w_1, w_2, \ldots, w_n\}$
Result: A CFG G
$V := \text{Sub}(D)$;
$I := \{[[u]] \mid u \in D\}$;
$P_L := \{[[a]] \to a \mid a \in \Sigma \cap V\}$;
$P_B := \{[[uv]] \to [[u]][[v]] \mid u, v, uv \in V\}$;
$P_U := \{[[u]] \to [[v]] \mid \exists (l,r) \wedge lur \in D \wedge lvr \in D\}$;
output $G = \langle \Sigma, V, I, P_L \cup P_B \cup P_U \rangle$

$c\}$, $P_B = \{[[ac]] \to [[a]][[c]], [[cb]] \to [[c]][[b]], [[acb]] \to [[ac]][[b]], [[acb]] \to [[a]][[cb]]\}$ *and* $P_U = \{[[c]] \to [[acb]], [[acb]] \to [[c]]\}$. *As can be verified this* CFG *defines the language* $\{a^n cb^n \mid n \geq 0\}$.

There are two natural ways to turn this grammar construction procedure into a learning algorithm. One is simply to apply this procedure to all of the available data. This will give a WBC-learner for the class of substitutable CFGs.

Alternatively since we can parse with the grammars, we can convert this into a GOLD learner, by only changing the hypothesis when the hypothesis is demonstrably too small. This means that once we have a weakly correct hypothesis the learner will no longer change its output. This simple modification gives a variant of the learner in Clark and Eyraud (2007). However this does not mean that this is a strong learner, since it may converge to a different hypothesis for different presentations of the same language. For example if a presentation of the language from Example 1 starts $\{a, acb, \ldots\}$ then the learner will converge in two steps to the grammar shown in Example 1. If on the other hand, the presentation starts $\{acb, aacbb, \ldots\}$ then it will also converge in two steps, but to a different, larger, grammar that includes nonterminals such as $[[aa]]$ and has a larger set of productions. This grammar is weakly equivalent to the former grammar, but it is not isomorphic or structurally equivalent, as it will assign a larger set of parses to strings like *aacbb*. It is more ambiguous. Indeed it is easy to see that this grammar will assign every possible binary branching structure to any string that is part of the set that the grammar is constructed from. And of course, the presentation could start with an arbitrarily long string—in which case the first grammar which it generates could be arbitrarily large.

5. The Syntactic Structure of Substitutable Languages

In this section we use a modification of Clark (2011) as the basis for our canonical grammars; in the case of substitutable languages the theory is quite simple so we will not present it in all its generality. Each nonterminal/syntactic category will correspond to a congruence class. With substitutable languages, we can show that the language itself, considered as a set of strings, has a simple intrinsic structure that can be used to define a particular finite grammar.

We start with the following definition:

Definition 7 *A congruence class X is* prime *if it is non-zero and non-unit and for any two congruence classes Y, Z such that $X = Y \cdot Z$ then either Y or Z is the unit. If a non-zero non-unit congruence class is not prime then we say it is* composite.

In other words a class is not prime if it can be decomposed into the concatenation of two other congruence classes. The stipulation that the unit and zero congruence classes are not prime is analogous to the stipulation that 1 is not a prime number. We will not give a detailed exposition of why the concept of a prime congruence class is important, but one intuitive reason is this. If we have nonterminals that correspond to congruence classes, and a congruence class N is composite, then that means that we can decompose N into two classes P,Q such that $N = PQ$. In that case we can replace every occurrence of N on the right hand side of a rule by the sequence $\langle P,Q\rangle$; assuming that P and Q can be represented adequately, nothing will be lost. Thus non-prime congruence classes can always be replaced by a sequence of prime congruence classes, and we can limit our attention to the primes which informally are those where "the whole is greater than the sum of the parts". More algebraically, we can think of the primes as representing the points where the concatenation operations in the free monoid and the syntactic monoid differ in interesting ways.

Example 2 *Consider the language $L = \{a^n cb^n \mid n \geq 0\}$. This language is not regular and therefore has an infinite number of congruence classes of which three are prime. The congruence classes are as follows:*

- $\{\lambda\}$ *is a congruence class with just one element; this is the unit congruence class which is not prime.*
- *The zero congruence class which consists of all strings that have empty distribution.*
- *L is a congruence class which is prime.*
- $[a] = \{a\}$ *is prime as is* $[b] = \{b\}$.
- *We also have an infinite number of congruence classes of the form $\{a^i\}$ for any $i > 1$. These are all composite as they can be represented as $[a] \cdot [a^{i-1}]$; similarly for $\{b^i\}$.*
- *Similarly we have classes of the form $[a^i c] = \{a^{i+j} cb^j \mid j \geq 0\}$ and $[cb^i] = \{a^j cb^{i+j} \mid j \geq 0\}$ which again are composite.*

L is not always prime as the following trivial example demonstrates.

Example 3 *Consider the finite language $L = \{ab\}$. This language has 5 congruence classes: $[a],[b],[ab],[\lambda]$ and the zero congruence class. The first 4 are all singleton sets. $[a]$ and $[b]$ are prime but $[ab] = \{ab\} = [a][b]$, and so L is not prime.*

Proposition 8 *For every $a \in \Sigma$, for any language L, if $[a]$ is non-zero and non-unit then $[a]$ is prime.*

Proof Let a be some letter in a language L and let $[a]$ be its congruence class. Suppose there are two congruence classes X,Y such that $XY = [a]$. Since $a \in [a]$, a must be in XY. Since we cannot split a string of length 1 into two non-empty strings, one of X and Y must be the unit. ■

We can now define the class of languages that we target with our learning algorithm.

Definition 9 *Let $\mathcal{L}_{sc}$ be the set of all languages which are substitutable, non-empty, do not contain λ and have a finite number of prime congruence classes.*

Given that there are substitutable languages which are not CFLs—the MIX language (Kanazawa and Salvati, 2012) being a good example—we need to restrict the class in some way. Here we consider only languages where there are a finite number of prime congruence classes. This implies, as we shall see later, that the language is a CFL. Every regular language of course has a finite number of primes as it has a finite number of congruence classes. Not all substitutable context-free languages have a finite number of primes, as this next example shows.

Example 4 *Consider the language* $L = \{c^i ba^i b \mid i > 0\} \cup \{c^i de^i d \mid i > 0\}$. *This is a substitutable context-free language. The distribution of* $ba^i b$ *is the single context* $\{(c^i, \lambda)\}$ *which is the same as that of* $de^i d$. *Therefore we have an infinite number of congruence classes of the form* $\{ba^i b, de^i d\}$, *each of which is prime.*

Definition 10 *A* prime decomposition *of a congruence class* X *is a finite sequence of one or more prime congruence classes* $\alpha = \langle X_1, \ldots, X_k \rangle$ *such that* $X = \bar{\alpha}$.

Clearly any prime congruence class X has a trivial prime decomposition of length one, namely $\langle X \rangle$. We have a prime factorization lemma for substitutable languages; we can rather pompously call this the 'fundamental lemma' by analogy with the fundamental lemma of arithmetic. This lemma means that we can represent all of the congruence classes exactly using just concatenations of the prime congruence classes.

Lemma 11 *Every non-zero non-unit congruence class of a language in* $\mathcal{L}_{sc}$ *has a unique prime factorisation.*

For proof see Lemma 33 in the appendix. Note that this is not the case in general for languages which are not substitutable, as the following example demonstrates.

Example 5 *Let* $L = \{abcd, apcd, bx\}$. *Note that L is finite but not substitutable since* $p \not\equiv b$. *Among the congruence classes are* $\{a\}, \{b\}, \{c\}$ $\{ab, ap\}$, $\{bc, pc\}$ *and* $\{abc, apc\}$. *Clearly* $\{ab, ap\}$, $\{bc, pc\}$ *are both prime but* $\{abc, apc\}$ *is composite and has the two distinct prime decompositions* $\{ab, ap\} \cdot \{c\}$ *and* $\{a\} \cdot \{bc, pc\}$.

If we restrict ourselves to languages in $\mathcal{L}_{sc}$ then we can assume without loss of generality that the nonterminals of the generating grammar correspond to congruence classes. In a substitutable language, a trim CFG cannot have a nonterminal that generates two strings that are not congruent. Similarly, if the grammar had two distinct nonterminals that generated congruent strings, we could merge them without altering the generated language.

Given that non-regular languages will have an infinite number of congruence classes, and that CFGs have by definition only a finite number of nonterminals, we cannot have one nonterminal for every congruence class. However in languages in $\mathcal{L}_{sc}$ there are only finitely many prime congruence classes, and since every other congruence class can be represented perfectly as a sequence of primes, it is sufficient to consider a grammar which has nonterminals that correspond to the primes. Therefore we will consider grammars whose nonterminals correspond only to the prime congruence classes of the grammar: we add one extra nonterminal S, a start symbol, which will not appear on the right hand side of any rule.

5.1 Productions

We now consider an abstract notion of a production where the nonterminals are the prime congruence classes.

Definition 12 *A* correct *branching production is of the form* $[\bar{\alpha}] \to \alpha$ *where* α *is a sequence of at least 2 primes and* $[\bar{\alpha}]$ *is a prime congruence class. A correct lexical production is one of the form* $[a] \to a$ *where* $a \in \Sigma$*, and* $[a]$ *is prime.*

Example 6 *Consider the language* $L = \{a^n cb^n \mid n \geq 0\}$*. This has primes* $[a], [c]$ *and* $[b]$*. The correct lexical productions are the three obvious ones* $[a] \to a$*,* $[b] \to b$ *and* $[c] \to c$*. The only correct branching productions have* $[c]$ *on the left hand side, and are* $[c] \to [a][c][b]$*,* $[c] \to [a][a][c][b][b]$ *and so on.*

Clearly in the previous example we want to rule out productions like $[c] \to [a][a][c][b][b]$ since the right hand sides are too long, and will make the derivation trees too flat. We want each production to be as simple as possible. Informally we say that the right hand side of the production $[a][a][c][b][b]$ is too long since there is a proper subsequence $[a][c][b]$ which generates strings in a prime congruence class, and should be represented just by the prime $[c]$.

Definition 13 *We say that a sequence of primes* α *is* pleonastic *(too long) if* $\alpha = \gamma\beta\delta$ *for some* γ, β, δ*, which are sequences of primes, such that* $|\gamma| + |\delta| > 0$*,* $[\bar{\beta}]$ *is a prime, and* $|\beta| > 1$*.*

Definition 14 *We say that a correct production* $N \to \alpha$ *is* pleonastic *if* α *is pleonastic. A correct production is* valid *if it is not pleonastic.*

Note that a pleonastic production by definition must have a right hand side of length at least 3.

For any string w in a prime congruence class where $w = a_1 \ldots a_n$, $a_i \in \Sigma$ we can construct a correct production $[w] \to [a_1] \ldots [a_n]$. Such productions may in general be pleonastic because there may be substrings that can be represented by prime congruence classes. From a structural perspective, the local trees derived from these productions are too shallow as they flatten out relevant parts of the structure of the string. Nonetheless we can find a set of valid productions that will generate the string w from the nonterminal $[w]$, as Lemma 18 below shows.

5.2 Canonical Grammars

We will now define canonical grammars for every language L in $\mathcal{L}_{sc}$. Note that for every language in $\mathcal{L}_{sc}$, L is a congruence class.

First of all we need the following lemma to establish that the grammar will be finite: see proof of Lemma 35 in the appendix.

Lemma 15 *If* $L \in \mathcal{L}_{sc}$ *then there are a finite number of valid productions.*

Definition 16 *Let* L *be some language in* $\mathcal{L}_{sc}$*. We define the following grammar,* $G_*(L)$*. The nonterminals are the prime congruence classes of* L*, together with an additional symbol* S*, which is the start symbol. Let* $\alpha(L)$ *be the unique prime decomposition of* L*. We define the set of productions,* P*, to have the following elements:*

- *the single production containing the start symbol:* $S \to \alpha(L)$,
- *all valid productions, of which there are only finitely many by Lemma 15,*
- *for each terminal symbol a that occurs in the language, the production* $[a] \to a$.

This is a unique CFG for every language in $\mathcal{L}_{sc}$. We now show that $G_*(L)$ generates L.

Lemma 17 *If $L \in \mathcal{L}_{sc}$ is a substitutable language, then for any prime congruence class X, $\mathcal{L}(G_*(L), X) \subseteq X$.*

Proof This is a simple induction on the length of the derivation. For $X \to a$, we know that $a \in X$ by construction. Suppose $X_i \stackrel{*}{\Rightarrow} u_i$ for all $1 \leq i \leq n$ and $X_0 \to X_1 \dots X_n$ is a production in the grammar. Then by the inductive hypothesis $u_i \in X_i$ and by the correctness of the production, $u_1 \dots u_n \in X_0$. ■

Lemma 18 *Suppose $X \to \alpha$ is a correct production. Then $X \stackrel{*}{\Rightarrow}_{G_*(L)} \alpha$.*

Proof By induction on the length of α. Base case: α is of length 2, in which case it cannot be pleonastic, and so $X \to \alpha$ is valid and in $G_*(L)$, and therefore $X \stackrel{*}{\Rightarrow} \alpha$. Inductive step: consider a correct production $X \to \alpha$ where α is of length k. If it is not pleonastic, then it is valid, and so $X \to \alpha$ is a production in $G_*(L)$, and so $X \stackrel{*}{\Rightarrow} \alpha$. Alternatively it is pleonastic and therefore $\alpha = \beta\gamma\delta$ where γ is the right hand side of a correct production, $Y \to \gamma$. Consider $X \to \beta Y \delta$, and $Y \to \gamma$. Both $\beta Y \delta$ and γ are shorter than α, and so by the inductive hypothesis $X \stackrel{*}{\Rightarrow} \beta Y \delta$ and $Y \stackrel{*}{\Rightarrow} \gamma$ so $X \stackrel{*}{\Rightarrow} \alpha$. So the lemma follows by induction. ■

Lemma 19 *Suppose X is a prime, and $w \in X$. Then $X \stackrel{*}{\Rightarrow}_{G_*(L)} w$.*

Proof If w is of length 1, then we have $X \to w$. Let $w = a_1 \dots a_n$ be some string of length $n > 1$. Let $\alpha = [a_1] \dots [a_n]$. So $X \to \alpha$ is a correct production. Therefore by Lemma 18 $X \stackrel{*}{\Rightarrow} \alpha$. Since we have the lexical rules $[a_i] \to a_i$ we can also derive $\alpha \stackrel{*}{\Rightarrow} w$. ■

Proposition 20 *For any $L \in \mathcal{L}_{sc}$, $\mathcal{L}(G_*(L)) = L$.*

Proof Suppose L has prime factorisation $A_1 \dots A_n$. S occurs on the left hand side of the single production $S \to A_1 \dots A_n$. Since $\mathcal{L}(G_*(L), A_i) = A_i$ by Lemmas 17 and 19, $\mathcal{L}(G_*(L), S) = A_1 \dots A_n = L$. ■

Definition 21 *We define $\mathcal{G}_{sc}$ to be the set of canonical context-free grammars for the languages in $\mathcal{L}_{sc}$:*

$$\mathcal{G}_{sc} = \{G_*(L) \mid L \in \mathcal{L}_{sc}\}.$$

Lemma 22 *$\mathcal{G}_{sc}$ is not redundant.*

Proof Suppose we have two weakly equivalent grammars G_1, G_2 in this class; then $G_1 = G_*(\mathcal{L}(G_1)) = G_*(\mathcal{L}(G_2)) = G_2$ and so they are isomorphic. ■

6. An Algorithm for Strong Learning

We now present a strong learning algorithm. We then demonstrate in Section 7 that for all grammars in $\mathcal{G}_{sc}$ the algorithm strongly converges in the SGOLD framework.

In outline, the algorithm works as follows; we accumulate all of the data that we have seen so far into a finite set D. We start by using Algorithm 1 to construct a CFG G^w which will be weakly correct for a sufficiently large input data. Using this observed data, together with the grammar which is used for parsing, we can then compute the canonical grammar for the language as follows.

1. We partition $\mathrm{Sub}(D)$ into congruence classes, with respect to our learned grammar G^w.
2. We pick the lexicographically shortest string in each class as the label we will use for the nonterminal.
3. We then test to see which of the congruence classes are prime.
4. Each class is decomposed uniquely into a sequence of primes.
5. A set of valid rules is constructed from the strings in the prime congruence classes.
6. We then eliminate pleonastic productions from this set of productions.
7. Finally, we return a grammar G^s constructed from these productions.

We can perform the first task efficiently, using the grammar and the substitutability property. Given that each string in $\mathrm{Sub}(D)$ occurs in the sample D, for each substring u we have some context (l,r) such that $lur \in D$. Given the substitutability condition, v is congruent to u iff $lvr \in \mathcal{L}(G_*)$. Under the assumption that the grammar is correct we can test this by seeing whether $lvr \in \mathcal{L}(G^w)$, using a standard polynomial parser, such as a CKY parser.

We now have a partition of $\mathrm{Sub}(D)$ into k classes $C_1,\ldots,C_k$. We pick the lexicographically shortest element of each class (with respect to $\prec$) which we denote by $u_1,\ldots,u_k$. Given a class, we want to test whether it is prime or not. Take the shortest element w in the class. Test every possible split of w into non-zero strings u,v such that $uv = w$. Clearly there are $|w|-1$ possible splits—for each split, identify the classes of u,v and test to see whether every element in the class can be formed as a concatenation of these two. If there is some string that cannot be split, then we know that the congruence class must be prime. If on the other hand we conclude that the class is not prime, we might potentially be wrong: we might for example think that $X = YZ$ simply because we have not yet observed one of the strings in $X \setminus YZ$. We present the pseudocode for this procedure in Algorithm 2.

For all of the non-prime congruence classes, we now want to compute the unique decomposition into primes. There are a number of obvious polynomial algorithms. We start by taking the shortest string w in a class; suppose it is of length n consisting of $a_1 \ldots a_n$. We convert this into a sequence of primes $[a_1]\ldots[a_n]$. We then greedily convert this into a unique shortest sequence of primes by checking every proper subsequence of length at least 2, and seeing if that string is in a prime congruence class. If it is then we replace that subsequence by the prime. We repeat until there are no proper subsequences that are primes. Alternatively we can use a shortest path algorithm. We create a graph which has one node for each $0,1,\ldots,n$. We create an arc from $i \to j$ if the substring spanning $[i,j]$ is prime. We then take the shortest path from 0 to n; and read off the sequence of

Algorithm 2 Testing for primality

Data: A set of strings X
Data: A partition of strings $\mathcal{X} = \{X_1, \ldots, X_n\}$, such that $\text{Sub}(X) \subseteq \bigcup \mathcal{X}$
Result: True or false
Select shortest $w \in X$;
for $u, v \in \Sigma^+$ such that $uv = w$ **do**
  $X_i \in \mathcal{X}$ is the set such that $u \in X_i$;
  $X_j \in \mathcal{X}$ is the set such that $v \in X_j$;
  if $X \subseteq X_i X_j$ **then**
    return false;
  end if
end for
return true;

primes by looking at the primes of the relevant segments. Note that since the lexical congruence classes are all prime, we know there will be at least one such path; since the language is substitutable we know this will be unique.

We then identify a set of valid productions. Every valid production will be of the form $N \to M\alpha$ where N, M are primes and α is a prime decomposition of length at least 1. For any given N, M there will be at most one such rule. Accordingly we loop through all triples of N and M, Q as follows: for each prime N, for each prime M, for each class Q, take α to be the prime decomposition of Q, and test to see if $N \to M\alpha$ is valid. We can test if it is correct easily by taking any shortest string u from M and any shortest string v from α and seeing if $uv \in N$; if it does then the rule is correct. Then we can test if it is valid by taking every proper prefix of $M\alpha$ of length at least two and testing if it corresponds to a prime. If no prefix does then the production is not pleonastic and is therefore valid.

For the lexical productions, we simply add all productions of the form $[a] \to a$ where $a \in \Sigma$. For the initial symbol S, we identify the unique congruence class of strings in the language X. If it is prime, then we add a rule $S \to X$. If it is not prime, and α is its unique prime decomposition then we add the rule $S \to \alpha$.

7. Analysis

We now proceed to the analysis of Algorithm 3, the learner $\mathcal{A}_{\text{SGOLD}}$. We want to prove three things: first that the algorithm strongly learns a certain class; secondly, that the algorithm runs in polynomial update time; finally that the algorithm converges rapidly, in the technical sense that it has a polynomially sized characteristic set.

We now are in a position to state our main result. We have defined a learning model, SGOLD, an algorithm A_{SGOLD} and a class of grammars $\mathcal{G}_{sc}$.

Theorem 23 A_{SGOLD} SGOLD-*learns the class of grammars* $\mathcal{G}_{sc}$.

In order to prove this we will show that for any presentation of a grammar in the class we will converge strongly to a grammar isomorphic to the canonical grammar. In what follows we suppose G_* is a grammar in $\mathcal{G}_{sc}$, and that $L_* = \mathcal{L}(G_*)$. For a grammar $G_* \in \mathcal{G}_{sc}$, we define $\chi(G_*)$ to be

Algorithm 3 $\mathcal{A}_{\text{SGOLD}}$ Strong Gold Learning Algorithm

Data: A sequence of strings $w_1, w_2, \ldots$
Data: Σ
Result: A sequence of CFGs $G_1, G_2, \ldots$
let $D := \varnothing$;
for $n = 1, 2, \ldots$ **do**
  let $D := D \cup \{w_n\}$;
  $\hat{G} = \hat{G}(D)$;
  Let C be the partition of $\text{Sub}(D)$ into classes;
  Let Pr be the set of primes computed using Algorithm 2;
  For each class N in C compute the prime decomposition $\alpha(N) \in Pr^+$;
  Let $V = \{[[N]] \mid N \in Pr\}$ be a set of nonterminals each labeled with the lexicographically shortest element in its class;
  Let S be a start symbol;
  $P_L = \{[[N]] \to a \mid [[N]] \in V, a \in \Sigma \cap N\}$;
  $P_I = \{S \to [[N]] \mid \exists w \in N, w \in D\}$;
  $P_B = \emptyset$;
  for $N \in Pr$, $M \in Pr$, $Q \in C$ **do**
    $R = (N \to M\alpha(Q))$;
    if R is correct and valid **then**
      $P_B = P_B \cup \{R\}$;
    end if
  end for
  output $G_n := \langle \Sigma, V \cup \{S\}, \{S\}, P_L \cup P_B \cup P_I \rangle$;
end for

a sufficiently large, yet polynomially bounded set of strings from $\mathcal{L}(G_*)$ such that when the input data includes this set, the weak grammar output will be correct (Clark and Eyraud, 2007) and which contains the shortest string in each prime congruence class.

Definition 24 *For a grammar $G = \langle \Sigma, V, I, P \rangle$ we define $\chi(G)$ as follows. For any $\alpha \in (\Sigma \cup V)^+$ we define $w(\alpha) \in \Sigma^+$ to be the smallest word, according to $\prec$, generated by α. Thus in particular for any word $u \in \Sigma^+$, $w(u) = u$. For each non-terminal $N \in V$ define $c(N)$ to be the smallest pair of terminal strings (l, r) (extending $\prec$ from Σ^* to $\Sigma^* \times \Sigma^*$, in some way), such that $S \stackrel{*}{\Rightarrow} lNr$. We now define the characteristic set $\chi(G_*) = \{lwr \mid (N \to \alpha) \in P, (l, r) = c(N), w = w(\alpha)\}$.*

We prove the correctness of the rest of the model under the assumption that the input data contains $\chi(G_*)$ and as a result that G^w is weakly correct: $\mathcal{L}(G^w) = \mathcal{L}(G_*)$. First, if G^w is correct, then the partition of $\text{Sub}(D)$ into congruence classes will be correct in the sense that two strings of $\text{Sub}(D)$ will be in the same class iff they are congruent.

Lemma 25 *Suppose $X_1, \ldots, X_n$ is a correct partition of $\text{Sub}(D)$ into congruence classes. Then if Algorithm 2 returns true when applied to X_i, then $[X_i]$ is in fact prime.*

Proof Suppose $[X_i]$ is not prime: then there are two congruence classes Y, Z such that $[X_i] = YZ$. Consider a string $w \in X_i$. There must be strings u, v such that $w = uv$ and $u \in Y, v \in Z$. Since

$w \in \text{Sub}(D)$, $u, v \in \text{Sub}(D)$. Since the partition of $\text{Sub}(D)$ is correct, there must be sets X_j, X_k such that $u \in X_j, v \in X_k$. Therefore, using again the correctness and the fact that $\text{Sub}(D)$ is substring closed, we have that $X_i \subseteq X_j X_k$, in which case Algorithm 2 will return false. ■

Lemma 26 *Suppose $X_1, \ldots, X_n$ is a correct partition of $\text{Sub}(D)$ into congruence classes, and $D \supseteq \chi(G_*)$. Then Algorithm 2 returns true when applied to X_i, iff $[X_i]$ is in fact prime.*

Proof X_i is a finite subset of $\text{Sub}(D)$, and we assume that all of the elements of X_i are in fact congruent. We already showed one direction, namely that if the algorithm returns true then $[X_i]$ is prime (Lemma 25). We now need to show that if $[X_i]$ is prime, then the algorithm correctly returns true. If $[X_i]$ is prime, then it will correspond to some nonterminal in the canonical grammar G_*, say N. There will be more than one production in G_* with N on the left hand side, and so by the construction of $\chi(G_*)$, and the correctness of the weak grammar, we will have at least one string from each production in $\text{Sub}(D)$, which means that since it is a correct partition the algorithm cannot find any pair of classes whose concatenation contains X_i. ■

As a consequence of this Lemma, we know that the algorithm will be able to correctly identify the set of primes of the language, and as a result will converge to the right set of nonterminals.

Proposition 27 *If the input data includes $\chi(G_*)$, then $G^s \cong G_*$.*

Proof We can verify that all and only the valid productions will be generated by the algorithm by the construction of the characteristic set.

Suppose $N \to X_1 \ldots X_n$ is a valid production in the grammar. Then by the construction of the characteristic set we will have a unique congruence class in the grammar corresponding to $[X_2 \cdots X_n]$. If $n > 2$ then this will be composite, and if $n = 2$ this will be prime, but in any event it will have a unique prime decomposition which will be exactly $\langle X_2, \ldots, X_n \rangle$, by Lemma 33. Therefore this production will be produced by the algorithm.

Secondly suppose the algorithm produces some production $N \to X_1, \ldots X_n$. We know that this will be valid since $X_2, \ldots X_n$ is a prime decomposition and is thus not pleonastic, and we tested all of the prefixes. We know that it will be correct, by the correctness of the weak learner and the fact that the congruence classes are correctly divided. It is easy to verify that the lexical and initial rules are also correctly extracted. ■

To conclude the proof of Theorem 23, we just need to observe that since the characteristic set includes the shortest element of each prime congruence class, and so the labels for each nonterminal will not change which means that the output grammars will converge exactly.

We now consider the efficiency of the algorithm. It is easy show that this algorithm runs in polynomial time in the size of the data set $\|D\|$, noting first that $|\text{Sub}(D)|$ is polynomial in $\|D\|$, and that as a result the grammars generated are all of polynomial size. Moreover the characteristic set has cardinality which is polynomial in the size of the grammar, and whose size is polynomially bounded in the thickness (Wakatsuki and Tomita, 1993).

S → **NT0**	**S** → **NT0**	**S** → **NT0**
NT1 → **b**	**NT3** → **open**	**NT2** → **NT2 NT0**
NT2 → **a**	**NT2** → **close**	**NT2** → **NT0 NT2**
NT0 → **c**	**NT4** → **neg**	**NT2** → **a**
NT0 → **NT2 NT0 NT1**	**NT4** → **NT0 NT1**	**NT0** → **NT0 NT0**
	NT0 → **a**	**NT0** → **NT2 NT1**
	NT0 → **b**	**NT0** → **NT1 NT2**
	NT0 → **c**	**NT1** → **b**
	NT0 → **NT3 NT4 NT0 NT2**	**NT1** → **NT1 NT0**
	NT1 → **and**	**NT1** → **NT0 NT1**
	NT1 → **iff**	
	NT1 → **implies**	
	NT1 → **or**	

Table 1: Output grammars for the three examples; on the left the grammar for $\{a^n cb^n \mid n \geq 0\}$, in the middlem the language of propositional logic, and on the right, the ambiguous grammar for $\{w \in \{a,b\}^+ \mid |w|_a = |w|_b\}$.

8. Examples

We have implemented the algorithm presented here.[2] We present the results of running this algorithm on small data sets that illustrate the properties of the canonical grammars for the learned languages. These examples are not intended to demonstrate the effectiveness of the algorithm but merely as illustrative examples to help the reader understand the representational assumptions, and as a result we have restricted ourselves to very simple languages which will be easy to understand. Nonterminals in the output grammar are either **S** for the start symbol or **NT** followed by a digit for the congruence classes that correspond to primes.

8.1 Trivial Context-Free Language

Consider the running example of $\{a^n cb^n \mid n \geq 0\}$. A characteristic set for this is just $\{c, acb\}$. Given this input data, we get the grammar shown on the left of Table 1. This defines the correct language; Figure 1 shows the parse trees for the three shortest strings in the language. This grammar is unambiguous so every string has only one tree.

8.2 Propositional Logic

Our next example is the language of sentential logic, with a finite number of propositional symbols. We have the alphabet $\{A_1, \ldots, A_k, (,), \neg, \vee, \wedge, \Rightarrow, \Leftrightarrow\}$. We would standardly define this language with the CFG: $S \to A_i$, $S \to (\neg S)$, $S \to (S \vee S)$, $S \to (S \wedge S)$, $S \to (S \Rightarrow S)$ and $S \to (S \Leftrightarrow S)$. Note that in this language the brackets are part of the object language not the meta-language—the algorithm does not know that they are brackets or what their function is. We replace them with other symbols in the experiment to emphasize this point. Thus the algorithm is given only flat sequences

2. A Java implementation will be made available on the author's website.

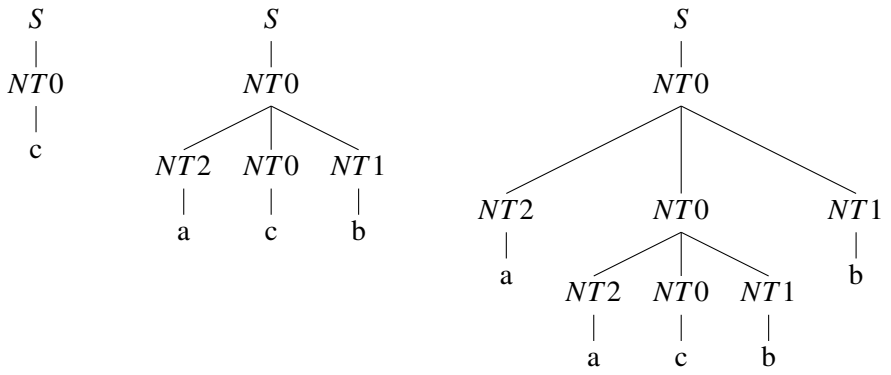


Figure 1: Example parse trees for the example $\{a^n cb^n \mid n \geq 0\}$.

of strings—there is implicitly structural information here, but the algorithm must discover it, as it must discover that the correct grammar is unambiguous. Sentential logic is an interesting example because it illustrates a case where the algorithm works but produces a different parse tree, but one that is still adequate for semantic interpretation. The canonical structure does not look like the ancestral tree we would see in a textbook presentation (Enderton, 2001).

Since $(\neg A)$ and $(A \vee A)$ are both in the language, $\neg \cong A\vee$, so the parse tree for $(A \vee B)$ will look a little strange: the canonical grammar has pulled out some more structure than the textbook grammar does: see Figure 2 for example trees. Nonetheless this is still suitable for semantic interpretation and the grammar is still unambiguous.

We fix some input data, replacing the symbols with strings to obtain input data of { *a, b, c, open a and b close, open a or b close, open a implies c close, open a iff c close, open neg a close* }. This produces the grammar shown in the middle of Table 1, which is weakly correct. This generates one tree for each string in the language as shown in Figure 2.

8.3 An Ambiguous Language

The next example is the language which consists of equal numbers of a's and b's in any order: $\{w \in \{a,b\}^+ \mid |w|_a = |w|_b\}$. We give the input data: $\{ab, ba, abab, abba, baba, bbaa\}$. The resulting grammar has 10 productions as shown on the right of Table 1.

In this case the grammar is ambiguous and the number of parses for each string varies, depending on properties of the string that are more complex than just the length. For example, the string *abab* has 5 parses, the string *abba* has 3 and the string *aabb* has only 2.

9. Discussion

Our goal in this paper is to take a small but theoretically well-founded step in a novel direction. This is not merely a new learning result but a new type of learning result: a *strong* learning result for a class of languages that includes non-regular languages. The main points of this paper are to define the learning model, and to establish that it is possible to obtain such results for at least some CFGs from positive strings alone. To the best of the author's knowledge this is the first non-trivial learning result of this type. There are of course trivial enumerative algorithms that can strongly learn any

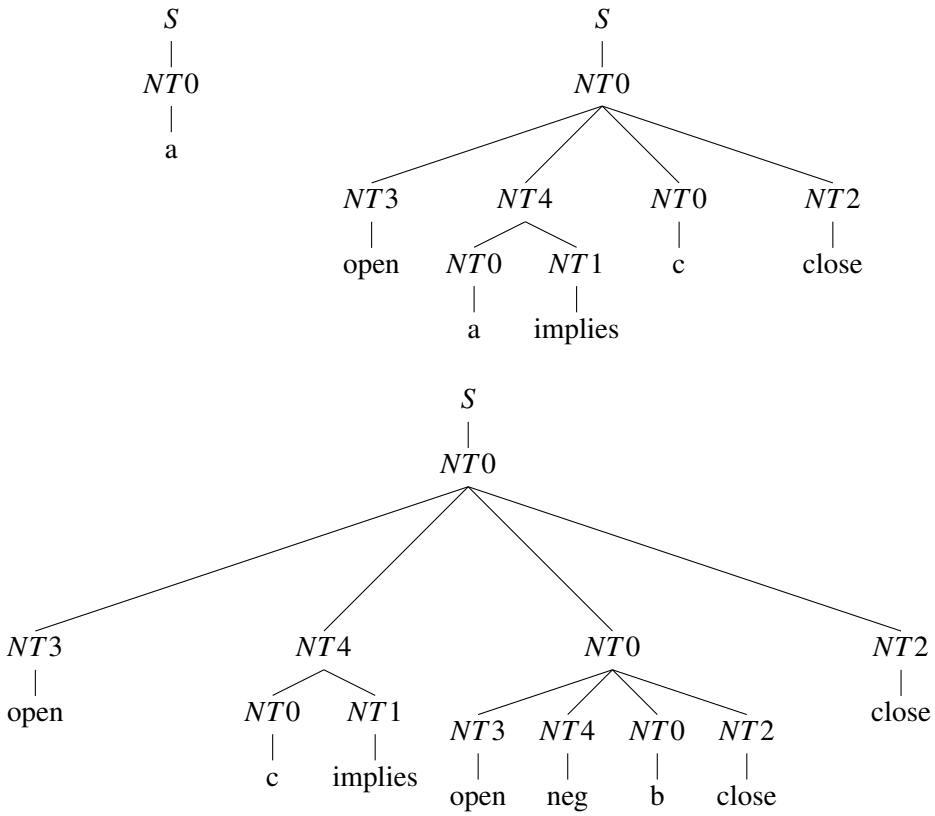


Figure 2: Example parse trees for the sentential logic example. Each example has only one parse tree.

non-redundant finite class of CFGs from positive data given a list of the elements of the class ordered by inclusion, and as mentioned before, the algorithm presented by Angluin (1982) can be viewed also as a strong learner for deterministic regular grammars. The Gold learning model is too onerous and as a result the class of languages that can be learned is very limited, but nonetheless includes some interesting natural examples as we showed in the previous section.

Strong learning is hard—accordingly we decompose it into two subproblems of rather different flavors. The first is a weak learning algorithm, and the second is a component that converts a weak learner to a strong learner; the latter component can be thought of as the computation of a canonical form. In general it will not be possible to compute a canonical form for an arbitrary grammar as this will be undecidable; however we may be able to do this for the grammars output by weak learners which will typically produce grammars in a restricted class.

In this paper, we have chosen to work using the simplest type of weak learner, and using only CFGs. The algorithm we have obtained therefore lacks some important features of natural language; notably lexical ambiguity and displacement. It also relies on an overly strong language theoretic closure property (substitutability) that natural languages do not satisfy. It is natural therefore to extend this in two ways. Firstly instead of using congruence classes as the basis for the nonterminals in the grammar, we can use syntactic concepts (Clark, 2013) which can be used to represent all

CFGs, and secondly we can move from CFGs to a much richer class of grammars—the class of well-nested multiple context-free grammars (Seki et al., 1991). The fundamental lemma is a nice technical result which simplifies the algorithm and the proof; however we will not have such a clean property in the case of larger classes of languages. Nonetheless we can extend the notion of a prime congruence class naturally to the richer mathematical structures that we need to model the more complex grammar formalisms required for natural language syntax.

Acknowledgments

I am grateful to Rémi Eyraud and Anna Kasprzik for helpful comments and discussion, and to Ryo Yoshinaka for detailed comments on an earlier draft. I would also like to thank the anonymous reviewers for their extremely constructive and helpful comments, which have greatly improved the paper. Any remaining errors are of course my own.

Appendix A.

This appendix contains the proofs of some technical lemmas that we use earlier that are not important from a learning theoretic point of view, but merely concern the algebraic properties of substitutable languages and their congruence classes. In all of the lemmas that follow, we assume we have a fixed language $L \in \mathcal{L}_{sc}$.

Lemma 28 *If X is a prime, and Y is a congruence class which is not equal to X, then there is a string in X which does not start with an element of Y.*

Proof Suppose every string in X starts with Y. Let x, x' be strings in X; then $x = yv$ and $x' = y'v'$ for some $y, y' \in Y$ and some other strings v, v'. Then $v \equiv v'$ by substitutability so $X = Y[v]$ and X is not prime. ■

Lemma 29 *Suppose $\alpha = A_1 \dots A_m$ and $\beta = B_1 \dots B_n$ are sequences of primes such that $\bar{\alpha} \supseteq \bar{\beta}$ then there is some j, $1 \leq j \leq n$ such that $A_1 \supseteq B_1 \dots B_j$.*

Proof If $B_1 = A_1$ then we are done. Alternatively pick some element $b_1 \in B_1$ which does not start with an element of A_1 (by Lemma 28). Now let w be some string in $B_2 \dots B_n$. Since $b_1 w \in \bar{\alpha}$ we must have some a_1, p_1 such that $a_1 = b_1 p_1$, where $a_1 \in A_1$. If $p_1 \in B_2$ then $B_1 B_2 \subseteq A_1$, so $j = 2$ and we are done. Otherwise take some element of B_2 that does not start with an element of $[p_1]$, say b_2. By the same argument we must have some $a_2 \in A_1$ and a p_2 such that $a_2 = b_1 b_2 p_2$, and where $b_2 p_2 \in [p_1]$. We repeat the process, and if we do not find some suitable j then we will have constructed a string in $\bar{\beta}$ which does not start with A_1 which contradicts the assumption that $\bar{\beta} \subseteq \bar{\alpha}$. Therefore there must be some j such that $B_1 \dots B_j \subseteq A_1$. ■

Lemma 30 *Suppose X is a prime, and α, β are strings of primes such that $X\bar{\alpha} \subseteq X\bar{\beta}$, where $X\bar{\beta} \subseteq \mathrm{Sub}(L)$, then $\bar{\alpha} \subseteq \bar{\beta}$.*

Proof Suppose $\alpha = A_1 \dots A_m$ and $\beta = B_1 \dots B_n$ are sequences of primes that satisfy the conditions of the lemma. Take some string in $\bar{\alpha}$, say a. Let x be a shortest string in X. xa is in the set $X\bar{\alpha}$ so we must have $xa = x'b$, for some $x' \in X, b \in \bar{\beta}$. Now x is the shortest string so either $x = x'$ and $a = b$ in which case the lemma holds, or $|x'| > |x|$ in which case we have $xcb = xa = x'b$, for some non-empty string c. So $xc = x'$ and x', x are both in X so $xc \equiv x$. Therefore $xcb \equiv xccb$ and so $b \equiv cb$, by substitutability. Now we can write b as a sequence of elements of β say $b = b_1 \dots b_n$, where $b_i \in B_i$. Since we have some context (l, r) such that $lbr \in L$ therefore $lcbr \in L$ by substitutability we will have $b_1 \equiv cb_1$ so $cb_1 \in B_1$ since it is a congruence class. This means that $cb \in \beta$ so $a \in \beta$ since $a = cb$. So $\bar{\alpha} \subseteq \bar{\beta}$. ■

An immediate corollary is this:

Lemma 31 *If X is a prime, and α, β are strings of primes such that $X\bar{\alpha} = X\bar{\beta}$, where $X\bar{\beta} \subseteq \mathrm{Sub}(L)$, then $\bar{\alpha} = \bar{\beta}$.*

Lemma 32 *If α and β are non-empty sequences of prime congruence classes such that $\bar{\alpha} = \bar{\beta} = [\bar{\alpha}]$, and $\bar{\alpha} \subseteq \mathrm{Sub}(L)$, then $\alpha = \beta$.*

Proof By induction on the length of the shortest string w in $\bar{\alpha}$. If this is 1 then clearly $\alpha = [w] = \beta$. Inductive step: suppose $\alpha = \langle A_1 \dots A_m \rangle$ and $\beta = \langle B_1, \dots B_n \rangle$. Since $\bar{\alpha} \subseteq \bar{\beta}$, we know by Lemma 29 that there must be some i such that $A_1 \dots A_i \subseteq B_1$ and similarly, since $\bar{\beta} \subseteq \bar{\alpha}$, there must be some j such that $B_1 \dots B_j \subseteq A_1$. Consider the shortest string $w \in \bar{\alpha}$. This means that $w = a_1 \dots a_m = b_1 \dots b_n$, where $a_k \in A_k, b_k \in B_k$. This means that all of the a_k, b_k are the shortest strings in their respective classes.

Suppose $a_1 \neq b_1$. Without loss of generality assume that $|a_1| > |b_1|$. This implies that $a_1 = b_1 s$, for some s. Now as we have seen, $A_1 \supseteq B_1 \dots B_i$, so $s \equiv b_2 \dots b_i$, by substitutability. If $|s| > |b_2 \dots b_i|$ then $a'_1 = s b_2 \dots b_i$ would be an even shorter element of A_1. If $|s| < |b_2 \dots b_i|$ then $b_1 s b_{i+1} \dots b_n$ would be a shorter element of $\bar{\beta}$ (using the fact that $\bar{\beta} = [\bar{\beta}]$. So $s = b_2 \dots b_i$ and $a_1 = b_1 \dots b_i$. This means that $a_2 \dots a_m = b_{i+1} \dots b_m$.

Pick an $a' \in A_1$ which does not start with an element of B_1 (which exists by Lemma 28). Consider $w' = a' a_2 \dots a_m$ which must also be equal to $b'_1 \dots b'_n$, where $b'_k \in B_k$ as before.

So a' must be a prefix of b'_1 which means that $a' a_2 \dots a_j = b'_1$ by substitutability and so $a_{j+1} \dots a_m = b'_2 \dots b'_n$. So $|b'_2 \dots b'_n| = |a_{j+1} \dots a_m| < |a_2 \dots a_m| = |b_{i+1} \dots b_n| < |b_2 \dots b_n|$, which is a contradiction since $b_2, \dots b_n$ are the shortest strings in $B_2 \dots B_n$. So $a_1 = b_1$ and $A_1 = B_1$. By Lemma 31 and by induction this means that $\alpha = \beta$. ■

We now prove the 'fundamental lemma' of substitutable languages.

Lemma 33 *Every non-zero non-unit congruence class has a unique prime factorisation.*

Proof We show that every congruence class can be represented as a product of primes; uniqueness then follows immediately by Lemma 32. Base case: the shortest string in X of length 1. (X is not the unit, so we know it is not of length 0). Then it is prime, and can be represented uniquely as a product of 1 prime, itself. Inductive step: suppose X is a congruence class whose shortest string is of length k. If X is prime, then again it is uniquely representable so suppose it is not prime, and there

is at least one decomposition into two congruence classes Y, Z. Y, Z must contain strings of length less than k and so by the inductive hypothesis, Y and Z are both decomposable into sequences of prime congruence classes, $Y = Y_1 \dots Y_i$ and $Z = Z_1 \dots Z_j$ so $X = Y_1 \dots Y_i Z_1 \dots Z_j$. ■

Lemma 34 *Suppose N is a prime and α, γ are nonempty sequences of primes such that $N \to \gamma\alpha$ is a valid production. Then α is the prime decomposition of $[\bar{\alpha}]$.*

Proof By induction on the length of α. The base case where α is of length 1 is trivial by the definition of a prime decomposition. Inductive step: Let β be the prime decomposition of $[\bar{\alpha}]$. Clearly $\bar{\alpha} \subseteq \bar{\beta}$ and so by Lemma 29 we know that there is some j such that $A_1 \dots A_j \subseteq B_1$. If $j > 1$ then this would mean that the rule was pleonastic and thus not valid, therefore $j = 1$ and so $A_1 = B_1$; the result follows by induction. ■

Lemma 35 *$G_*(L)$ only has a finite number of valid productions.*

Proof Let n is the number of primes in the language L. Suppose we have two valid productions $N \to A\alpha$ and $N \to A\beta$, where N, A are primes and α, β are sequences of primes. Therefore by Lemma 34 $\alpha = \beta$, which means that there can be at most one production for each pair of primes N, A; therefore the total number of branching productions is at most n^2. ■

References

D. Angluin. Inference of reversible languages. *Journal of the ACM*, 29(3):741–765, 1982.

R.C. Berwick, P. Pietroski, B. Yankama, and N. Chomsky. Poverty of the stimulus revisited. *Cognitive Science*, 35:1207–1242, 2011.

L. Blum and M. Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28(2):125–155, 1975.

J. Case and C. Lynes. Machine inductive inference and language identification. *Automata, Languages and Programming*, pages 107–115, 1982.

A. Clark. A language theoretic approach to syntactic structure. In Makoto Kanazawa, András Kornai, Marcus Kracht, and Hiroyuki Seki, editors, *The Mathematics of Language*, pages 39–56. Springer Berlin Heidelberg, 2011.

A. Clark. The syntactic concept lattice: Another algebraic theory of the context-free languages? *Journal of Logic and Computation*, 2013. doi: 10.1093/logcom/ext037.

A. Clark and R. Eyraud. Polynomial identification in the limit of substitutable context-free languages. *Journal of Machine Learning Research*, 8:1725–1745, August 2007.

T. Cohn, P. Blunsom, and S. Goldwater. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096, 2010.

F. Drewes and J. Högberg. Learning a regular tree language from a teacher. In Zoltán Ésik and Zoltán Fülöp, editors, *Developments in Language Theory*, pages 279–291. Springer Berlin Heidelberg, 2003.

H. Enderton. *A Mathematical Introduction to Logic*. Academic press, 2001.

S. Ginsburg. *The Mathematical Theory of Context-Free Languages*. McGraw-Hill, New York, 1966.

E. Mark Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

D. Hsu, S. M. Kakade, and P. Liang. Identifiability and unmixing of latent parse trees. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1520–1528, 2013.

M. Kanazawa and S. Salvati. MIX is not a tree-adjoining language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 666–674. Association for Computational Linguistics, 2012.

D. López, J.M. Sempere, and P. García. Inference of reversible tree languages. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 34(4):1658–1665, 2004.

P.H. Miller. *Strong Generative Capacity: The Semantics of Linguistic Formalism*. CSLI Publications, Stanford, CA, 1999.

D. Osherson, M. Stob, and S. Weinstein. *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists*. MIT Press, first edition, 1986.

T. Petrie. Probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 40(1):97–115, 1969.

R.C. Read and D.G. Corneil. The graph isomorphism disease. *Journal of Graph Theory*, 1(4): 339–363, 1977.

Y. Sakakibara. Learning context-free grammars from structural data in polynomial time. *Theoretical Computer Science*, 76(2):223–242, 1990.

Y. Sakakibara. Efficient learning of context-free grammars from positive structural examples. *Information and Computation*, 97(1):23 – 60, 1992.

R. E. Schapire. A brief introduction to boosting. In *Proceedings of 16th International Joint Conference on Artifical Intelligence*, pages 1401–1406, 1999.

H. Seki, T. Matsumura, M. Fujii, and T. Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):229, 1991.

M. Wakatsuki and E. Tomita. A fast algorithm for checking the inclusion for very simple deterministic pushdown automata. *IEICE TRANSACTIONS on Information and Systems*, 76(10): 1224–1233, 1993.

K. Wexler and P. W. Culicover. *Formal Principles of Language Acquisition*. MIT Press, Cambridge, MA, 1980.

R. Yoshinaka. Identification in the Limit of k-l-Substitutable Context-Free Languages. In Alexander Clark, François Coste, and Laurent Miclet, editors, *Grammatical Inference: Algorithms and Applications*, pages 266–279. Springer Berlin Heidelberg, 2008.

R. Yoshinaka. Efficient learning of multiple context-free languages with multidimensional substitutability from positive data. *Theoretical Computer Science*, 412(19):1821 – 1831, 2011.

R. Yoshinaka. Integration of the dual approaches in the distributional learning of context-free grammars. In Adrian-Horia Dediu and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications*, pages 538–550. Springer Berlin Heidelberg, 2012.

R. Yoshinaka and M. Kanazawa. Distributional learning of abstract categorial grammars. In Sylvain Pogodalla and Jean-Philippe Prost, editors, *Logical Aspects of Computational Linguistics*, pages 251–266. Springer Berlin Heidelberg, 2011.

V.N. Zemlyachenko, N.M. Korneenko, and R.I. Tyshkevich. Graph isomorphism problem. *Journal of Mathematical Sciences*, 29(4):1426–1481, 1985.

Journal of Machine Learning Research 14 (2013) 3561-3589 Submitted 6/12; Published 12/13

Classifying With Confidence From Incomplete Information

Nathan Parrish PARRISH.NATHAN@GMAIL.COM
Applied Physics Laboratory
Johns Hopkins University
Laurel, MD 20723, USA

Hyrum S. Anderson HANDER@SANDIA.GOV
Sandia National Laboratories
Albuquerque, NM 87123, USA

Maya R. Gupta MAYAGUPTA@GOOGLE.COM
1225 Charleston Rd
Google Research
Mountain View, CA 94025, USA

Dun Yu Hsiao DYHSIAO@U.WASHINGTON.EDU
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-4322, USA

Editor: Kevin Murphy

Abstract

We consider the problem of classifying a test sample given incomplete information. This problem arises naturally when data about a test sample is collected over time, or when costs must be incurred to compute the classification features. For example, in a distributed sensor network only a fraction of the sensors may have reported measurements at a certain time, and additional time, power, and bandwidth is needed to collect the complete data to classify. A practical goal is to assign a class label as soon as enough data is available to make a good decision. We formalize this goal through the notion of reliability—the probability that a label assigned given incomplete data would be the same as the label assigned given the complete data, and we propose a method to classify incomplete data only if some reliability threshold is met. Our approach models the complete data as a random variable whose distribution is dependent on the current incomplete data and the (complete) training data. The method differs from standard imputation strategies in that our focus is on determining the reliability of the classification decision, rather than just the class label. We show that the method provides useful reliability estimates of the correctness of the imputed class labels on a set of experiments on time-series data sets, where the goal is to classify the time-series as early as possible while still guaranteeing that the reliability threshold is met.

Keywords: classification, sensor networks, signals, reliability

1. Introduction

In many applications there is a cost associated with collecting or computing the features necessary to classify a test sample. For example, in medical applications, there are costs to subjecting a patient to additional tests. Or, in distributed networks, the cost of aggregating all of the test data is power or bandwidth. In many applications, there is simply a CPU or bandwidth cost to getting and processing

raw data to produce a full set of classification features. Thus, it is desirable to know if one can make a good decision without collecting all of the test data. Specifically, we wish to guarantee that a decision made from incomplete test data has a high probability of being the same decision that would be made given the complete test data.

In this paper, we focus on answering the question "With probability at least equal to τ, will the classification decision from incomplete data be the same as that which would be made from the complete data?" Our approach also makes it possible to answer the related question, "If we classify based on the current incomplete data, what is the probability that the class decision will be the same as classifying from the complete data?"

First, we propose optimal and practical decision rules for classifying incomplete data. In Sections 3, 4, and 5 we provide the details on how to efficiently and accurately implement the proposed practical decision rule for classifiers that use linear or quadratic discriminants, such as linear support vector machines and linear or quadratic discriminant analysis (LDA and QDA). In Section 6, we review related work on classifying with missing features and related work on early classification of time-series data. Experiments in Section 7 show that the proposed incomplete decision rule consistently provides enhanced reliability over the state of the art in classifying incomplete data. We further discuss the results and some open questions in Section 8.

This paper significantly extends our prior work in the conference paper (Anderson et al., 2012), where we tackled the same problem but proposed a more conservative decision rule. In this paper, we propose a more tractable decision rule, show how it can be used with different kinds of classifiers, show that our approach can be applied to different features, and provide substantially more analysis and experimental results.

2. Incomplete Decision Rules

Let $\hat{g}(x)$ be a classifier function that assigns a class label to test sample x. However, suppose that at test time we do not have x, but instead have some incomplete information given as a vector z. We wish to classify z if it gives us enough information about x to make a good decision, otherwise, we delay making a decision until we have more information. To that end, we consider decision rules that answer the question: "Can we classify z and know that we meet some minimum probability threshold of making the same decision that we would make on x?" We use the term *reliability* to mean the probability that the class label assigned to z matches that assigned to x.

To estimate reliability, we model the classification features derived from the complete data as a random variable X, where X is jointly distributed with the random variable Z modeling the incomplete data. Given a desired reliability $\tau \in [0,1]$ and a realization of the incomplete information z, an ideal incomplete decision rule is to classify as class g if

$$\begin{aligned} P(\hat{g}(X) = g | Z = z) &= \int_{x \text{ s.t. } \hat{g}(x)=g} p(x|z)\, dx \\ &\geq \tau, \end{aligned} \tag{1}$$

and otherwise to wait for more information. Figure 2 illustrates this rule.

The ideal rule given in (1) could be checked in several ways. A straightforward check would be to compute the integral directly and see if it is greater than or equal to τ. An alternative check that we find easier to approximate is to consider all sets A in the domain of X such that $P(X \in A | Z = z) \geq \tau$,

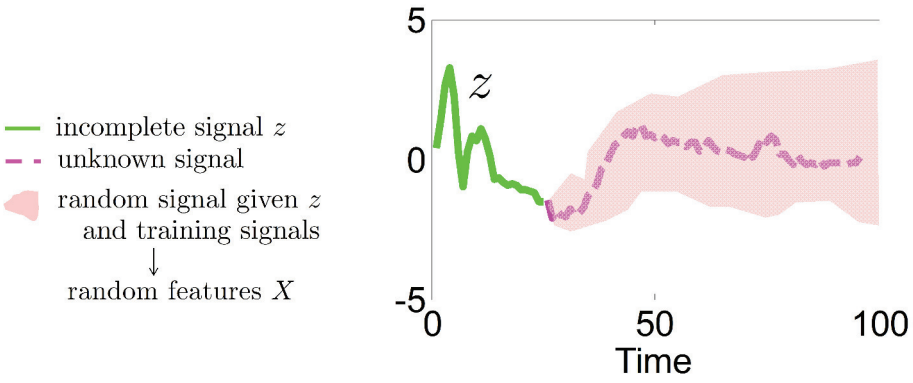


Figure 1: In this example, the available information is the incomplete time signal z, shown in green. Assuming the complete signal is iid with the training signals, the complete signal can be treated as a random signal (illustrated in pink), implying a conditional density on the complete signal's classification features, $p(x|z)$. Given $p(x|z)$ we can check whether or not one can make a reliable classification.

and see if $\hat{g}(x)$ maps all x in one such set to a single class g. In general, we expect both these checks to be computationally intractable.

We propose that a more conservative, but computable, incomplete decision rule is to classify as class g if

$$\hat{g}(x) = g \text{ for all } x \in A \text{ for some set } A \text{ such that } P(X \in A|Z = z) \geq \tau. \tag{2}$$

Rule (2) differs from (1) in that only one set A that contains at least τ measure of X must be checked. This rule is more conservative than (1) because it does not check all sets A, and thus (1) could be satisfied without (2) being satisfied (but not vice-versa).

Implementing the proposed rule given in (2) requires three steps. First, we must estimate the conditional density $p(x|z)$. Second, we must construct an appropriate set A, and third, we must check if the rule is satisfied. We first discuss the construction of a set A in Section 3, and show that our construction only requires estimates of the first and second conditional moments of X. Then in Section 4, we show how rule (2) can be efficiently checked for classifiers that have linear or quadratic class discriminant functions. We delay the discussion of how to estimate the necessary moments of X until Section 5.

3. Defining a Set A that Contains Measure τ of X

To implement the incomplete decision rule (2), one must be able to construct a set A that contains at least τ measure of X given z. In this section we propose three ways to construct such a set A. Figure 3 compares these three constructions.

3.1 Chebyshev Construction for Set A

Suppose that we estimate only the first and second conditional moments of X, but make no assumptions about the distribution other than that it has finite first and second moments. Then a set A can

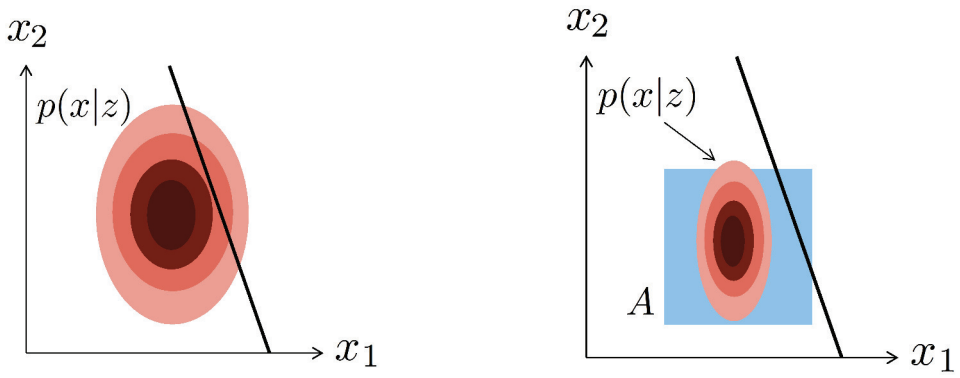


Figure 2: Comparison of the ideal and proposed conservative but computable decision rule. **Left:** A two-dimensional feature space and a linear class decision boundary. The probability mass of X lies mostly to the left of the decision boundary. For desired reliability values τ that are smaller than the probability mass of X that falls to the left of the decision boundary, the ideal incomplete decision rule would choose to classify based on the incomplete information z. **Right:** The entire probability mass of X falls on one side of the decision boundary, and thus the ideal incomplete decision rule would choose to classify rather than wait, for every value of τ. However, our computable incomplete decision rule constructs a set A that captures a fraction τ of the mass of X and requires that entire set A to lie on one side of the decision boundary. For the choice of A shown here in blue, the set A crosses the decision boundary, and thus the computable decision rule would choose to wait for more information before classifying.

be constructed using the multidimensional Chebyshev inequality, which states that for $X \in \mathbb{R}^d$ with known mean m and covariance R, and any $\alpha > 0$:

$$P\left((X-m)^T R^{-1}(X-m) \leq \alpha^2\right) \geq 1 - \frac{d}{\alpha^2}.$$

Thus to satisfy $P(X \in A|Z=z) \geq \tau$, define

$$A = \left\{ x \text{ s.t. } (x-m)^T R^{-1}(x-m) \leq \frac{d}{1-\tau} \right\}. \tag{3}$$

The set A defined by (3) is non-empty for $\tau \in (-\infty, 1]$, although $\tau \leq 0$ does not give a useful bound for the incomplete classifier reliability.

3.2 Naive Bayes Constructions for Set A

The Chebyshev construction given in the previous section can be overly conservative, as it makes no assumptions about the conditional distribution of X other than a finite mean and covariance. If we assume more about the distribution, we can define a smaller constraint set A that results in a less conservative decision rule, and therefore earlier classification for the same reliability requirement τ.

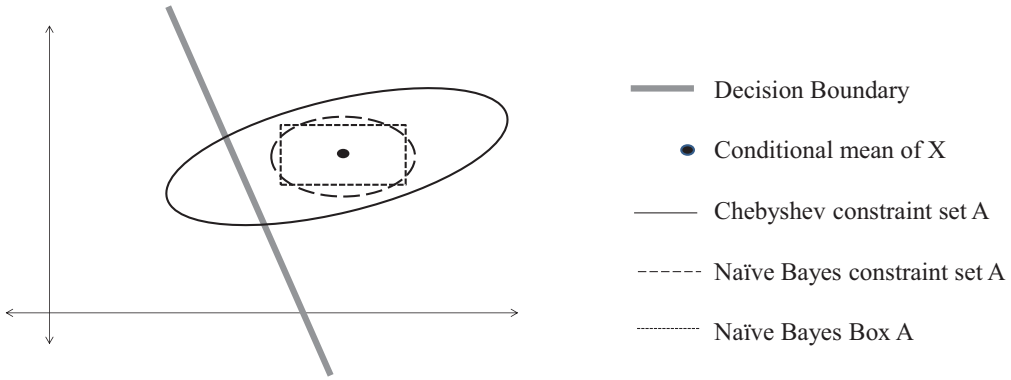


Figure 3: Example sets that contain mass τ of the conditional p.d.f. of X formed by the three different construction methods for A proposed Section 3.

For example, if we assume that the conditional distribution is Gaussian,[1] then a quadratic set A that covers τ measure of X is

$$A = \left\{x \text{ s.t. } (x-m)^T R^{-1}(x-m) \leq \text{erf}(\tau)\right\}, \tag{4}$$

where one must compute the inverse cdf to determine the value $\text{erf}(\tau)$ to achieve a set A with measure τ. For a multi-dimensional Gaussian, computing the inverse cdf for (4) is non-trivial. We can simplify (4) by making the conservative näive Bayes assumption that the components of X are independent, and thus R is diagonal. Then the quadratic function in (4) becomes $\sum_{\ell=1}^{d}\left(\frac{x(\ell)-m(\ell)}{\sqrt{R(\ell,\ell)}}\right)^2$. Under the independent Gaussian assumption, $\sum_{\ell=1}^{d}\left(\frac{X(\ell)-m(\ell)}{\sqrt{R(\ell,\ell)}}\right)^2$ is a chi-squared random variable with d degrees of freedom; thus, the $\text{erf}(\tau)$ function in (4) is easily computed using the inverse cdf of a chi-squared random variable.

A related option is to force the set A to be a box. Again, make the näive Bayes assumption that elements of X are independent, then $p(x|z) = \prod_{\ell=1}^{d} p(x(\ell)|z)$. Therefore, we can define a set

$$A = \{x \text{ s.t. } x(\ell) \in [m(\ell) - s_\tau(\ell), m(\ell) + s_\tau(\ell)] \; \forall \; \ell = 1, ..., d\}, \tag{5}$$

where s_τ is a vector defining the width of the box in each dimension such that the total measure of the box is τ. In this paper, we implement this constraint by assigning each dimension equal measure $\tau^{1/d}$ while assume that each marginal distribution $X(\ell)$ is Gaussian.

The two options (4) and (5) make the same two assumptions about the conditional distribution of X, but (4) finds the ellipsoidal footprint of the Gaussian that has measure τ, while (5) treats the dimensions completely independently, giving each of the marginals measure $\tau^{1/d}$.

1. The Gaussian assumption is often justified by a central limit theorem argument, a maximum entropy argument, or a simplicity argument.

4. Efficient Solutions for Linear or Quadratic Discriminants

In this section, we show that the incomplete data classification rule (2) with the constraint sets A proposed in Section 3 can be computed efficiently for classifiers of the form

$$\hat{g}(x) = \arg\max_{g} f_g(x), \tag{6}$$

where $f_g(x)$ is a linear or quadratic discriminant function for the g^{th} class, and according to (6), the classifier assigns x to the class with the maximum discriminant. For example, the linear support vector machine (SVM) has a linear discriminant, while the quadratic discriminant analysis (QDA) classifier has a quadratic discriminant (Hastie et al., 2001).

Nearest-neighbor classifiers using an Euclidean (or Mahalanobis) distance have a discriminant that over the set $x \in A$ requires taking the minimum of a set of quadratic discriminants:

$$f_g(x) = \min_{x_i : y_i = g} (x - x_i)^T (x - x_i).$$

An optimal method for checking the incomplete decision rule (2) for this discriminant is an open question. A conservative reliability decision can be made by treating each sample as its own class in (6). That is, let $f_i(x) = (x - x_i)^T (x - x_i)$, solve (6) for the resulting quadratic discriminant, and then classify as the class y_i. A computationally simpler approach (but one that is not strictly conservative), is to only consider each class's nearest neighbor to the posterior mean, which produces one quadratic discriminant per class. In experiments, we do something similar to the latter approach using a local QDA classifier.

We begin with the two-class problem, and then show how this rule can be extended to multi-class problems.

4.1 Two-class Problems

We first consider a two-class problem, where the set of class labels is $\mathcal{G} = \{1, 2\}$. Let $f_1(x)$ and $f_2(x)$ be the discriminants for classes one and two, and define

$$f(x) = f_2(x) - f_1(x).$$

We can define an equivalent classifier to (6) using only $f(x)$ by noting that $f(x) = 0$ defines the decision boundary between classes 1 and 2. Therefore, classification rule (6) is equivalent to

$$\hat{g}(x) = \begin{cases} 1 & \text{if } f(x) \leq 0 \\ 2 & \text{if } f(x) > 0. \end{cases}$$

Then the proposed incomplete data decision rule (2) is implemented:

$$\hat{g}(z) = \begin{cases} 1 & \text{if } \max_{x \in A} f(x) \leq 0 \\ 2 & \text{if } \min_{x \in A} f(x) > 0 \\ \text{no decision} & \text{otherwise.} \end{cases} \tag{7}$$

Note that the decision rule (7) is dependent on the incomplete data through the dependence of A on z. The three different conditions in (7) are shown for a quadratic discriminant (and hence quadratic decision boundary) and a quadratic construction of the set A in Figure 4.

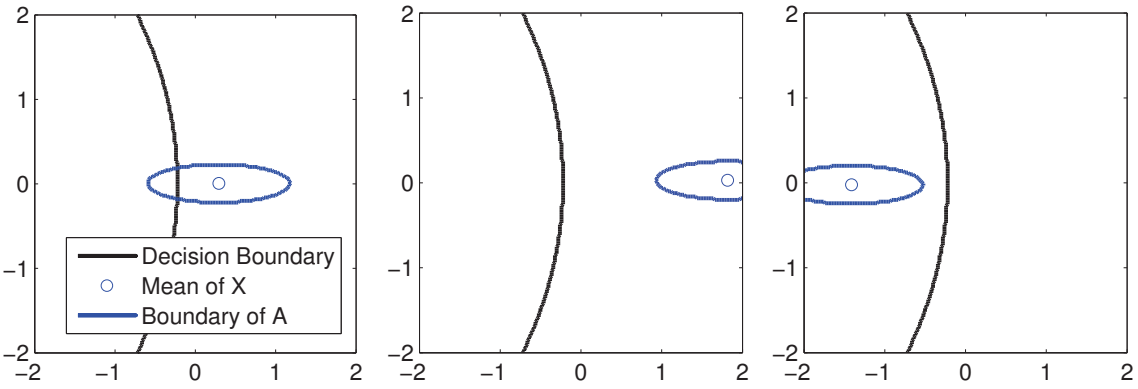


Figure 4: Three different scenarios for incomplete data classification. In the leftmost plot, the classifier withholds making a decision. In the center and rightmost plots, A lies completely on a single side of the decision boundary, so the classifier assigns a label to the incomplete data.

4.1.1 LINEAR DISCRIMINANTS

In order to efficiently check (7), we must be able to efficiently compute the maximum and minimum of $f(x)$ over the set $x \in A$. If $f_1(x)$ and $f_2(x)$ are linear discriminants, then $f(x)$ is also linear. Coupled with a quadratic set A, such as the Chebyshev or näive Bayes quadratic sets A given in Section 3, finding the maximum and minimum are the linear programs with quadratic constraints:

$$\max_{x \in A} f(x) = \max_{x} \beta^T x + b \qquad (8)$$
$$\text{s.t. } (x-m)^T R^{-1} (x-m) \leq \delta$$

$$\min_{x \in A} f(x) = \min_{x} \beta^T x + b \qquad (9)$$
$$\text{s.t. } (x-m)^T R^{-1} (x-m) \leq \delta.$$

These optimizations have closed-form solutions:

Proposition 1: *The solutions to (8) and (9) are, respectively*

$$\max_{x \in A} f(x) = \beta^T m + \sqrt{\delta} \parallel R^{1/2} \beta \parallel_2 + b$$
$$\min_{x \in A} f(x) = \beta^T m - \sqrt{\delta} \parallel R^{1/2} \beta \parallel_2 + b.$$

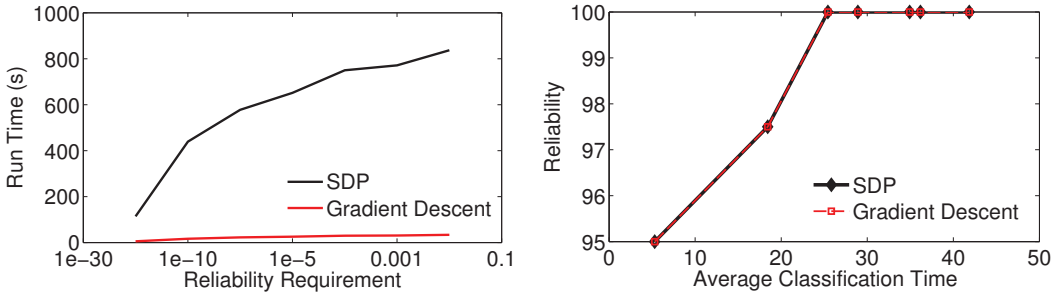


Figure 5: The left figure shows the time required by the SDP vs gradient descent solutions. The right figure verifies that the solution for the methods is identical.

For a linear set A such as the näive Bayes box constraint set given in (5), the maximum and minimum are:

$$\max_{x\in A} f(x) = \max_{x} \beta^T x + b \qquad (10)$$
$$\text{s.t. } m(\ell) - s_\tau(\ell) \le x \le m(\ell) + s_\tau(\ell)\ \forall\ \ell = 1,\ldots,d$$

$$\min_{x\in A} f(x) = \min_{x} \beta^T x + b \qquad (11)$$
$$\text{s.t. } m(\ell) - s_\tau(\ell) \le x \le m(\ell) + s_\tau(\ell)\ \forall\ \ell = 1,\ldots,d.$$

The solution of (10) is $\beta^T m + |\beta^T| s_\tau + b$, and the solution of (11) is $\beta^T m - |\beta^T| s_\tau + b$.

4.1.2 QUADRATIC DISCRIMINANTS

If the class discriminant functions are quadratic, then $f(x) = f_2(x) - f_1(x)$ will also be quadratic and, thus, can be written

$$f(x) = (x-v)^T V (x-v) + b. \qquad (12)$$

Since (12) is the difference of two quadratics, V will generally be indefinite even if $f_2(x)$ and $f_1(x)$ are both positive semi-definite.

First consider finding the maximum and minimum of (12), as required by the incomplete decision rule (7), over a quadratic constraint set A. Since V is indefinite, this is a non-convex optimization problem; however, strong duality holds for finding the minimum or maximum of a quadratic function subject to quadratic constraints (see, e.g., Boyd and Vandenberghe, 2008). The dual problem is a semi-definite program (SDP), and can therefore be solved using convex optimization such as an interior point method. However, in our experiments, we found the SDP solution to be prohibitively slow. Therefore, we instead propose to use the two-step gradient descent approach described in Appendix B. Martinez (1994) showed that there is at most one local non-global solution to this non-convex problem. Also, since we need only know if the minimum or maximum of $f(x)$ is less than or greater than zero, we can often stop the gradient descent before convergence. Figure (5) shows a run-time comparison between the SDP solution solved using Sedumi and the gradient-descent solution.

Now consider finding the maximum and minimum of (12) over the box set A. An efficient solution is obtained by first performing a change of variables that diagonalizes V. Define $y = V^{1/2}x$ and $w = V^{1/2}v$, then $f(x) = f(y) = \| y - w \|^2 + b$. After this change of variables, we can greatly simplify the maximum and minimum computations required by the incomplete decision rule (7) by making the näive Bayes assumption on the random variable $Y = V^{1/2}X$ as opposed to on X. Defining the mean of Y as $m_y = V^{1/2}m$,

$$\max_{x \in A} f(x) = \max_{y \in A} \| y - w \|^2 + b \tag{13}$$

$$\min_{x \in A} f(x) = \min_{y \in A} \| y - w \|^2 + b, \tag{14}$$

with

$$A = \{ y \text{ s.t. } y(\ell) \in [m_y(\ell) - s_\tau(\ell),\ m_y(\ell) + s_\tau(\ell)],\ \forall\ \ell = 1, ..., d \},$$

where the $s_\tau(\ell)$ are determined by the inverse cdf of $Y(\ell)$.

After this change of variables, the y that maximizes (13) is found by assigning each $y(\ell)$ to the edge of the box that maximizes the distance from $w(\ell)$. Similarly, the y that minimizes (14) assigns $y(\ell) = w(\ell)$ if $w(\ell) \in [m_y(\ell) - s_\tau(\ell),\ m_y(\ell) + s_\tau(\ell)]$. Otherwise, $y(\ell)$ is assigned to the edge of the box that minimizes the distance to $w(\ell)$.

4.2 Multi-class Classifiers

We now extend the results of the previous section to multi-class classifiers. For multi-class classifiers, the classification rule (6) can be expressed:

$$\hat{g}(x) = c \ \text{ if } \ f_c(x) - f_h(x) \geq 0 \ \text{ for all } \ h \neq c.$$

The proposed incomplete data classification rule (2) can be written:

$$\hat{g}(z) = \begin{cases} c & \text{if } \min_{x \in A} f_c(x) - f_h(x) \geq 0 \text{ for all } h \neq c \\ \text{no decision} & \text{otherwise.} \end{cases} \tag{15}$$

That is, classify z as class c if the set A lies completely within the decision region for some class c, and do not decide at the requested reliability if the set A straddles a decision boundary.

If there are G total classes, then (15) implies $2\binom{G}{2}$ possible checks of the form $\min_{x \in A} f_c(x) - f_h(x) \geq 0$. However, we show in the next section that regardless of the construction of set A, one must compute at most $2(G-1)$ of these checks. Furthermore, if the set A contains the posterior mean m (as it does in all of our proposed constructions for A), then a decision can be made with at most $G-1$ checks using this two-step procedure:
Step 1 - Guess: Let $c = \arg\max_g f_g(m)$.
Step 2 - Check: Sequentially check if $\min_{x \in A} f_c(x) - f_h(x) \geq 0$ for $h = 1, 2, \ldots, G,\ h \neq c$. If the check fails for any h, stop, and output the result *no decision*. If the check holds for all h, then classify early as class c.

4.3 General Multiclass Decision Process

We provide a provably efficient multi-class decision process for arbitrary constructions of the constraint set A. We say that class c *dominates* class h and that class h is *dominated* by c if $f_c(x) - f_h(x) \geq 0$ for all $x \in A$. If neither class dominates the other one, then the two classes are called *tied*. To classify the incomplete data z early as class c, class c must dominate all other classes.

4.3.1 PROPOSED DECISION PROCESS

Initialize: Begin with all G classes labelled `candidate`.
Compare: Choose any two classes c and h that are labelled `candidate` and check if $\min_{x\in A} f_c(x) - f_h(x) \geq 0$. If yes, then label h as `dominated`. If no, then perform a second check to see if $\min_{x\in A}\ f_h(x) - f_c(x) \geq 0$, and if so then label c as `dominated`, and otherwise label both classes as `tied`. Continue this process until fewer than two classes are labelled `candidate`. If no classes remain that are labelled `candidate`, then output *no decision*. If one class is labelled `candidate`, then proceed to the *Final Comparison*.
Final Comparison: Check if the last class labelled `candidate` dominates every class labelled `tied`. If yes, classify the incomplete data as the class labelled `candidate`, if no, output *no decision*.
Proposition 2: *The above decision process correctly determines the dominating class or that there is no dominating class.*
Proposition 3: *Given G classes, the above decision process requires at most $2(G-1)$ minimum problem calculations evaluations, and at least $G - \lfloor G/2 \rfloor$ pairwise evaluations.*

5. Estimation of the Complete Test Data Distribution

In order to construct the sets A in Section 3, we must estimate the mean m and covariance R of the complete test data X. We do this by leveraging the incomplete information about the test signal that is currently available along with the prior knowledge of the structure of the test signal gained from the training data using the standard assumption that the training and test features are IID. We present two estimation methods: 1) joint Gaussian estimation, and 2) Gaussian mixture model (GMM) estimation. These approaches are similar to those used in missing feature imputation, for example in speech recognition as described by Raj and Stern (2005). However, our approach differs from that of missing feature imputation in that the latter constructs only a point estimate of the unknown data, whereas we construct estimates of the mean and covariance of the unknown data.

5.1 Joint Gaussian Estimation

For joint Gaussian estimation, we assume that the complete data X is distributed jointly Gaussian with the incomplete data Z. Therefore, the model is

$$\begin{bmatrix} X \\ Z \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \bar{x} \\ \bar{z} \end{bmatrix}, \begin{bmatrix} \Sigma_{x,x} & \Sigma_{x,z} \\ \Sigma_{z,x} & \Sigma_{z,z} \end{bmatrix} \right). \tag{16}$$

We estimate the model parameters in (16) from the training data. The mean and covariance parameters of X conditioned on the realization of the partial information $Z = z$ are

$$m = \hat{\bar{x}} + \hat{\Sigma}_{x,z}\hat{\Sigma}_{z,z}^{-1}(z - \hat{\bar{z}})$$
$$R = \hat{\Sigma}_{x,x} - \hat{\Sigma}_{x,z}\hat{\Sigma}_{z,z}^{-1}\hat{\Sigma}_{z,x}.$$

5.2 GMM Based Estimation

We assume that the joint distribution of the complete data, X, and the incomplete data, Z, is a Gaussian mixture model, where the elements of the Gaussian mixture are the class-conditional

distributions. Under these assumptions the model is

$$\begin{bmatrix} X \\ Z \end{bmatrix} \sim \sum_{g \in \mathcal{G}} w(g) P\left(\begin{bmatrix} X \\ Z \end{bmatrix} \Big| \, g \right), \tag{17}$$

where $w(g)$ is the weight of the class g Gaussian and

$$P\left(\begin{bmatrix} X \\ Z \end{bmatrix} \Big| \, g \right) = \mathcal{N}\left(\begin{bmatrix} \bar{x}_g \\ \bar{z}_g \end{bmatrix}, \begin{bmatrix} \Sigma_{x,x}(g) & \Sigma_{x,z}(g) \\ \Sigma_{z,x}(g) & \Sigma_{z,z}(g) \end{bmatrix} \right).$$

We can again estimate the parameters of the model (the means, covariances, and weights), from the training data.

Define

$$\begin{aligned} m_g &= \hat{\bar{x}}_g + \hat{\Sigma}_{x,z}(g)\hat{\Sigma}_{z,z}^{-1}(g)(z - \hat{\bar{z}}_g), \\ R_g &= \hat{\Sigma}_{x,x}(g) - \hat{\Sigma}_{x,z}(g)\hat{\Sigma}_{z,z}^{-1}(g)\hat{\Sigma}_{z,x}(g), \\ p(g|z) &= p(G = g \,|\, Z = z) \\ &= \frac{w_g p(Z = z \,|\, G = g)}{\sum_{h \in \mathcal{G}} w_h p(Z = z \,|\, G = h)}. \end{aligned}$$

Given a realization $Z = z$, we can compute the mean m of X as:

$$m = \mathrm{E}[X|z] = \sum_{g \in \mathcal{G}} \mathrm{E}[X, G \,|\, z] = \sum_{g \in \mathcal{G}} m_g p(g|z).$$

Furthermore, as shown in Appendix C, the covariance of X is

$$R = \sum_{g \in \mathcal{G}} p(g|z) \left(R_g + m_g m_g^T \right) - \sum_{q \in \mathcal{G}} \sum_{h \in \mathcal{G}} m_q m_h^T p(q|z) p(h|z).$$

6. Related Work

We detail the related work in early classification and missing features, then we contrast the proposed with optimal stopping, feature selection, online and incremental learning, and sequential hypothesis ratio testing.

6.1 Other Early Classification Work

Xing et al. (1998) considered the problem of making an early prediction on time-series data that matches that of a full length one nearest-neighbor classifier. Suppose that the labelled training data set is $\{(x_i, g_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$. Their approach, called early classification on time-series (ECTS), is motivated by the idea of the *minimum prediction length* (MPL) of a training time-series x_i. Define $x_i(1:t) \in \mathbb{R}^t$ to be the first t samples of x_i. Furthermore, define, $\mathrm{RNN}(x_i(1:t))$ to be the reverse nearest neighbors of $x_i(1:t)$ which is the set of training samples that choose x_i to be their nearest neighbor at time t. The MPL of x_i is the smallest time index k such that for all $k \leq \ell \leq d$ the following holds $\mathrm{RNN}(x_i(1:\ell)) = \mathrm{RNN}(x_i(1:d)) \neq \emptyset$. By this definition, the MPL is the smallest time index at which the reverse nearest neighbors of x_i do not change as the rest of the time-series is

revealed. At test time, a training point x_i can be used to assign a label to a test sample $x(1:t)$ once $t \geq \text{MPL}(x_i)$, the minimum prediction length of x_i.

The authors found that the above procedure was too conservative; therefore, they proposed a slightly modified way to find the MPL for ECTS. They first clustered the training data using a hierarchical clustering method and then selected the MPL for each training time-series depending on its cluster membership. They also introduced a parameter to control the earliness of their approach called *minimum support*—a ratio that varies between zero and one, with zero resulting in the earliest classifier. However, the minimum support parameter is different from our τ parameter in that it does not provide an explicit guarantee on the reliability of the early decision.

Xing et al. cite Rodriguez and Alonso (2002) as the only existing study mentioning early classification on time-series data. Rodriguez and Alonso (2002) propose to classify a time-series using a *literal* based classifier, where a literal is a descriptor describing what happens during a specified interval of the time-series. For example, the literal *increases* would be set to one if the time-series increases during the specified interval, and would be set to zero otherwise. The authors mention that for early classification of time-series some of the literals will not yet have a value because the interval that they are measured in has not occurred yet. The authors propose to omit these literals from the classifier in order to classifier early.

6.2 Related Work on Missing and Noisy Features

Another related body of work is imputing (estimating) missing features. If missing features occur in the training data, then standard methods of classifier training cannot be used. One method of dealing with this problem, called single imputation, is to fill in the missing features with their estimated values. The missing features can be estimated using a multivariate regressor that is trained using the subset of training data with no missing features. Schafer and Graham (2002) and Rogier et al. (2006) review missing feature methods for training data.

When features are missing in the test data, there are three standard options (see, e.g., Saar-Tsechansky and Provost, 2007): imputing a point estimate for the missing features, imputing a distribution for the missing features, and the reduced-models approach. For the reduced models approach, classifiers are trained for each set of potentially missing information (Friedman et al., 1996; Schuurmans and Greiner, 2007; Saar-Tsechansky and Provost, 2007). Here, we do impute a distribution over the missing features (conditioned on the given information about the test sample and the training data statistics), but rather than just use that distribution to predict the best class label, we use the distribution to measure the reliability of a classification decision with the incomplete data. Thus, our contributions are in-part complementary to imputation methods, and different methods than the ones we used in Section 5 can be easily substituted into the proposed approach.

If features are noisy rather than missing, then estimating the clean feature values can improve test accuracy. This problem arises, for example, in automatic speech recognition (ASR) systems when the test signal is masked by noise (Cooke et al., 2001; Raj et al., 2004; Raj and Stern, 2005). Raj and Stern (2005) compare MAP estimates for noisy features in ASR systems using Gaussian and GMM based estimators with models similar to those that we describe in Section 5.

6.3 Optimal Stopping Rules

Quoting Ferguson (2001), "The *theory of optimal stopping* is concerned with the problem of choosing a time to take a given action based on sequentially observed random variables in order to maxi-

mize an expected pay-off or to minimize an expected cost." While the high-level goal is the same, the optimal stopping perspective requires specification of misclassification costs and delay costs, which are often difficult to specify. Given such costs, an optimal stopping rule approach would attempt to estimate the probability of each class given the current incomplete information, and determine the expected costs of making a decision or waiting.

6.4 Feature Selection

A related problem in classification is to determine the best subset of features to use in classification. For example, the classic *forward selection method* sequentially adds in features based on their marginal value. Different stopping rules have been proposed to decide when to stop sequentially adding the features (Costanza and Afifi, 1979). Generally stopping rules are not applicable to the problem we focus on because they assume that all increasing sets of features can be compared, rather than that one only has the incomplete set of features and must make a decision. In addition, stopping rules are based only on the training data statistics, and from our perspective are strictly suboptimal in that they do not consider the current incomplete information.

6.5 Online and Incremental Learning

In this paper we assume that a fixed set of training data is given, and that incremental features of a test sample become available. These assumptions differ from the usual set-up of online learning (also known as incremental learning), which assumes that incrementally more training data becomes available to train the classifier over time (e.g., Pang et al., 2005; Dredze et al., 2008; Crammer and Singer, 2003). Also assuming the online learning set-up, Fu et al. (2005) propose a stopping rule for deciding when enough training samples have been received to classify with confidence.

6.6 Sequential Hypothesis Testing

The sequential probability ratio test (SPRT) (Wald, 1947) is a greedy alternate to the proposed work, designed for use with probabilistic models of two hypotheses. In the context of binary classification, and a generative model $p(y|x_k)$, it accumulates the log-likelihood ratio:

$$S_k = S_{k-1} + \log p(y_1|x_k) - \log(y_2|x_k), \tag{18}$$

and if S_k exceeds a preset threshold t_1, the signal would be called for class 1, and if S_k goes below a preset negative threshold t_2, the signal would be called for class 2. The thresholds are set to achieve desired error levels on class 1 and class 2 respectively.

Armitage (1950) expanded SPRT for the multi-hypothesis case and applied it to linear discriminant analysis classification (in which each class is assumed to be drawn from a distribution with the same covariance matrix) for a different problem than the one treated here: given a sequence of iid samples from one class, he prescribed how to use SPRT to give a rule for how and when to determine the class.

A key difference between the proposed approach and the SPRT approach is that (18) is greedy: new features do not change the contribution to the log-likelihood already made by previous features, which stems from the standard SPRT assumption that successive observations are independent. But the classifiers we consider in this paper are not trained to consider the features independently. Further, we assume correlations between the features in order to estimate a probability distribution over the unknown part of the feature vector, which we use to define a constraint set.

Data set	Time-series Length	Number of Classes	Training Samples	Test Samples
Chlorine Concentration	166	3	467	3840
Italy Power Demand	24	2	67	1029
Face (All)	131	14	560	1690
Medical Images	99	10	381	760
Non-Invasive Fetal ECG 1	750	42	1800	1965
Non-Invasive Fetal ECG 2	750	42	1800	1965
Starlight Curves	1024	3	1000	8236
Swedish Leaf	128	15	500	625
Synthetic Control	60	6	300	300
Two Patterns	128	4	1000	4000
U Wave Gesture Library X	315	8	896	3582
U Wave Gesture Library Y	315	8	896	3582
U Wave Gesture Library Z	315	8	896	3582
Wafer	152	2	1000	6174
Yoga	426	2	300	3000

Table 1: Time-series Data Sets

7. Experiments

Section 7.1 details the data sets, experimental set-up, and classifiers used. We first compare the proposed methods to construct sets A of measure τ, reported in Section 7.2, and the proposed estimation methods for the moments of $P_{X|z}$, reported in Section 7.3. Then in Section 7.4, we show that applying a dimensionality reduction method can greatly reduce the computation needed at test time. Lastly, we compare our recommended reliable classifier to other approaches to early classification.

Research-grade code and the experimental data sets are available to download.[2]

7.1 Experimental Set-up and Details

We demonstrate performance using all of the time-series data sets available on the *UCR Time-Series Classification and Clustering Page* (Keogh et al., 2006) that have at least five hundred test samples and at least 15 training examples per class when this paper was written. We use the given training and test splits, so all results can be reproduced. We also use the Synthetic Control data set from this repository, a data set of Gaussian data that has only three hundred test samples, to further illustrate the differences between the constraint sets and estimation methods that we have described for the proposed incomplete decision rule. Table 1 gives details for the used data sets.

The time-series classification experiments are performed as follows. The test data set consists of n sampled time-series vectors and corresponding labels $\{x_i, g_i\}_{i=1}^n$, with $x_i \in \mathbb{R}^d$ and $g \in \mathcal{G}$. At time t, the incomplete data for the i^{th} test time-series is $z_i \in \mathbb{R}^t$, the first t samples of x_i. At each time t we check the proposed incomplete decision rule and classify z_i if the reliability condition is met for τ. We plot results for a set of choices of τ.

2. The code and data sets can be downloaded at `http://www.mayagupta.org/publications/Early_Classification_For_Web.zip`.

	Local QDA			Linear SVM		
	Chebyshev	Quadratic	Box	Chebyshev	Quadratic	Box
Synthetic Control	1.8	0.7	0.4	0.4	0.3	0.3
Medical Images	27.1	2.7	1.4	1.9	1.0	0.8
Two Patterns	12.45	2.0	1.0	1.8	0.5	0.3

Table 2: Average test time per sample, in seconds, for the three different constraint sets.

Let $t_i(\tau)$ be the minimum time at which the i^{th} test signal can be classified with reliability constraint τ, and let $\hat{g}(z_i(\tau))$ be the class label assigned to z_i at this time. We measure the test reliability as $\frac{1}{n}\sum_{i=1}^{n} \mathrm{I}(\hat{g}(z_i(\tau)) = \hat{g}(x_i))$, where $\hat{g}(x_i)$ is the label assigned to the complete data and $\mathrm{I}(\cdot)$ is one if the argument is true and zero otherwise. We also measure the average classification time as the mean of the $t_i(\tau)$. Ideally, we would like to classify with the smallest average classification time while still meeting reliability requirement τ.

We perform incomplete classification experiments with two different discriminant classifiers. The first classifier is local QDA (Garcia et al., 2010). Local QDA learns the mean and covariance for the class g discriminant function for test point x, $f_g(x)$, by estimating them using the k nearest class g training points to test point x. We choose $k \in \{1, 2, 4, 8, 16, 32, 64, 128\}$ by cross-validation on the training data. In our implementation of local QDA, we use a diagonal covariance matrix, and we regularize the covariance estimate by adding $10^{-4}\mathrm{I}$, where I is the identity matrix. Since we do not have the complete data x, we instead estimate the mean and covariance for $f_g(x)$ by finding the nearest class g neighbors to the mean of X. The second classifier that we use is a linear SVM which we implement using LibSVM (Chang and Lin, 2011) with default settings.

7.2 Comparison of Construction of Sets of Measure τ

We first compare the three set construction methods proposed Section 3, the Chebyshev set (3), the Gaussian näive Bayes quadratic set (4), and the Gaussian näive Bayes box set (5).

We vary the reliability parameter between four values $\tau = [0.001, 0.1, 0.25, 0.9]$, and we perform prediction using the jointly Gaussian model (16). Figure 6 plots the results for the Synthetic Control, Medical Images, and Two Patterns data sets. In all cases, the empirical reliability rate exceeds the reliability requirement τ. Additionally, these plots verify that the Chebyshev set is the most conservative, as it waits the longest to classify the test data, and the näive Bayes quadratic set is the least conservative.

Table 2 compares the average testing time per test sample for the three different constraint sets when $\tau = 0.9$. This table shows that the näive Bayes box set is the least computationally complex, followed by the näive Bayes quadratic set, and finally the Chebyshev set.

7.3 Comparison of Estimation Methods

In this section we compare the performance of reliable incomplete classification using jointly Gaussian estimation (16) to that using GMM estimation (17). We use the same classifiers and values for τ as given in the previous section.

Figure 7 plots the average classification time vs. test reliability for the jointly Gaussian and GMM estimation methods using the näive Bayes quadratic constraint set. The figure shows that on the Synthetic Control and Medical Images data sets, the GMM method dominates the jointly Gaus-

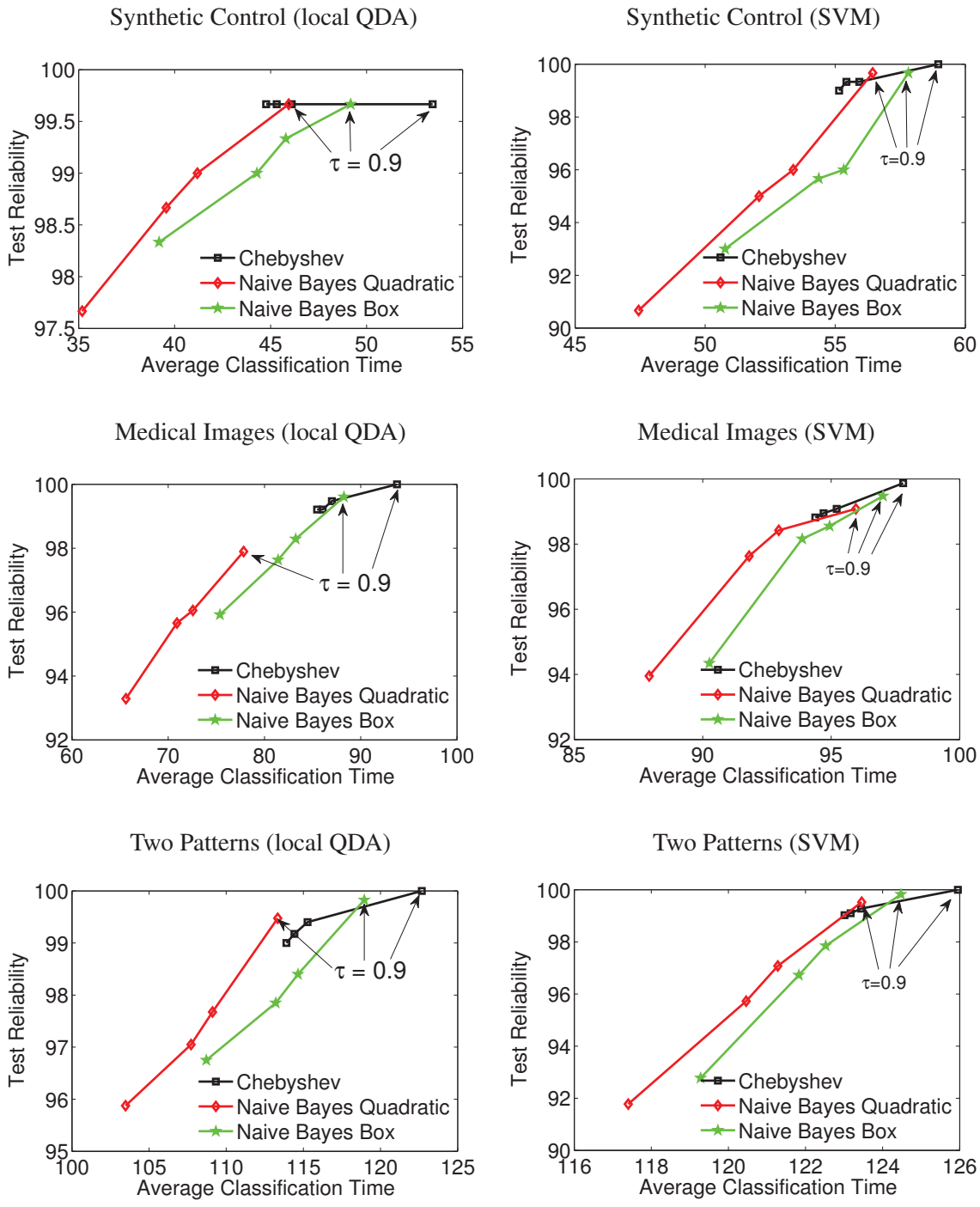


Figure 6: Average classification time vs test reliability for local QDA (left column) and linear SVM (right column) using jointly Gaussian prediction. The symbols correspond to choices of $\tau \in \{0.001, 0.1, 0.25, 0.9\}$.

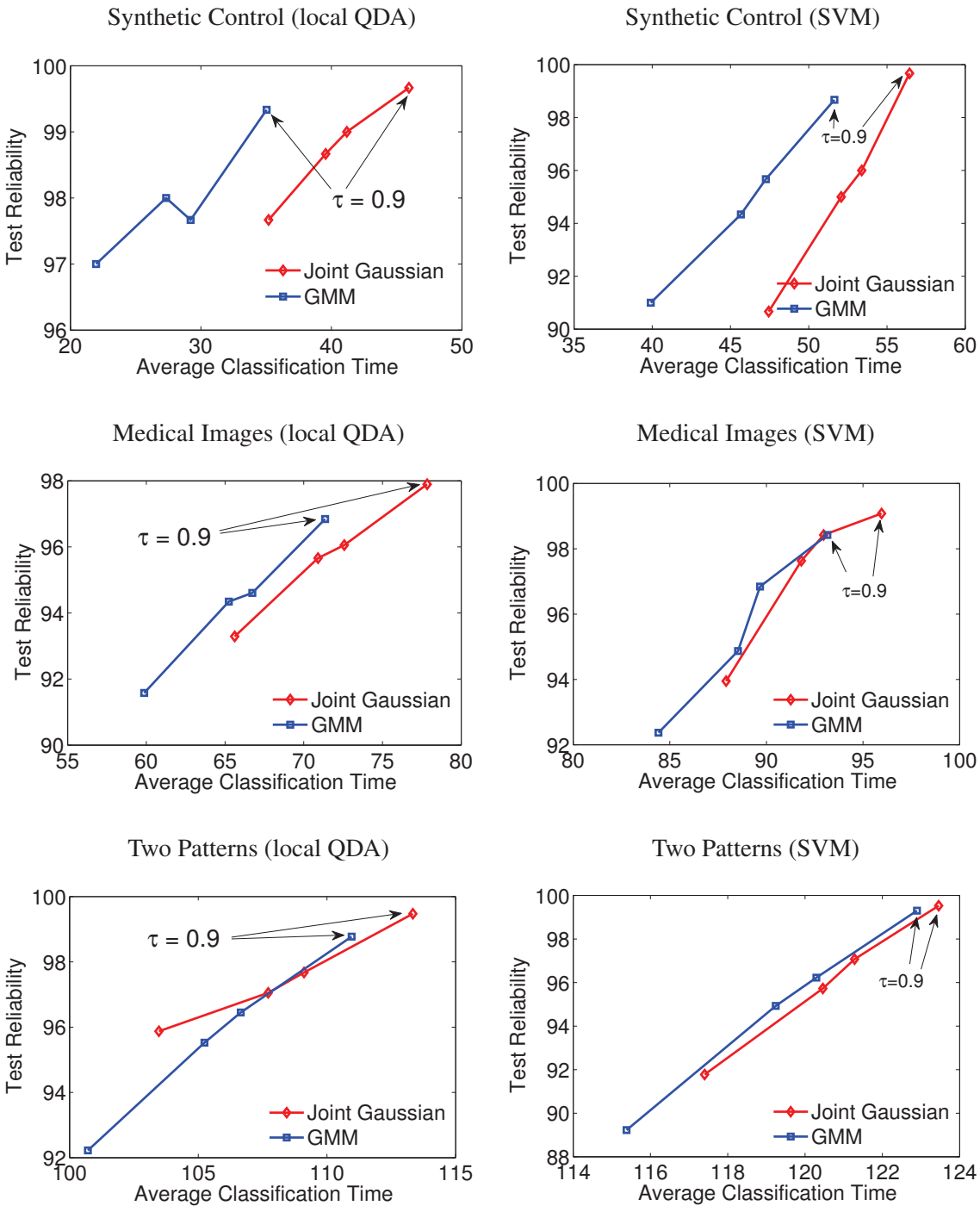


Figure 7: Average classification time vs test reliability for local QDA (left column) and linear SVM (right column) using the näive Bayes quadratic constraint set with τ varied between $[0.001, 0.1, 0.25, 0.9]$.

	Local QDA		Linear SVM	
	Joint Gaussian	GMM	Joint Gaussian	GMM
Synthetic Control	0.7	0.9	0.3	0.7
Medical Images	2.7	5.5	1.0	2.7
Two Patterns	2.0	3.4	0.5	0.8

Table 3: Average test time per sample, in seconds, for the two different estimation methods.

sian over all τ values for both classifiers. On the Two Patterns data set with local QDA classification, the GMM method is not uniformly better than the jointly Gaussian method.

Table 3 compares the total testing time of the two approaches, and as expected, the GMM method requires more test time than the simpler jointly Gaussian method.

7.4 Dimensionality Reduction Features

An advantage of our reliable incomplete classification approach is that it can use any features derived from the data for which we can estimate the mean and covariance. As an example alternative to using the time-series samples as the features, we select a smaller feature set by first preprocessing the time-series using supervised linear dimensionality reduction. Linear dimensionality reduction finds a matrix $B \in \mathbb{R}^{\ell \times d}$, $\ell < d$ that maps the data from d-dimensional to ℓ-dimensional space. Supervised dimensionality reduction uses the label information in the training data to find a reduced space where the data is also separated by class. In the context of incomplete data classification, the complete data becomes the vector $Bx \in \mathbb{R}^{\ell}$ as opposed to $x \in \mathbb{R}^{d}$.

Linear dimensionality reduction can provide two advantages over classifying the time-series features. First, it can diminish the impact of noisy or non-discriminative features in the time-series data, thus providing increased accuracy. Second, reducing the number of features reduces the computational complexity. For a time-series with d samples, there are $d-t$ unknown samples at time t. Thus, if we simply use the time-series samples as the features for classification, the optimization problem that the reliable incomplete classifier must solve has $d-t$ free variables. For a long time series, this can cause the computational complexity to become extreme when t is small. However, performing linear dimensionality reduction reduces the number of unknowns to ℓ which can greatly reduce the number of variables in the optimization for reliable classification.

We use local discriminative Gaussian (LDG) dimensionality reduction (Parrish and Gupta, 2012) to learn B. We choose LDG dimensionality reduction because 1) it can separate multi-modal data, 2) the solution is fast, requiring only a maximal eigenvalue decomposition, and 3) it has been shown to work well even when few training samples are provided and the input dimensionality is large. Furthermore, we can choose the best input dimensionality by performing cross-validation on the training data set to find a reduced space that is both small and accurate. Table 4 shows the dimensionality of the training data after LDG dimensionality reduction. The table also compares the testing time required to perform reliable local QDA classification with the näive Bayes quadratic constraint set with jointly Gaussian estimation at time $t = 1$ with and without LDG dimensionality reduction. On the data sets with more than three hundred time-series samples, using LDG dimensionality reduction results in an orders of magnitude decrease in the testing time.

Data set	Time-series length	Number of LDG features	Test time at t=1 (ms)	LDG test time at t=1 (ms)
Chlorine Concentration	166	42	76	4
Italy Power Demand	24	2	2	1
Face (All)	131	30	40	2
Medical Images	99	11	18	2
Non-Invasive Fetal ECG 1	750	30	6,107	4
Non-Invasive Fetal ECG 2	750	23	5,789	3
Starlight Curves	1024	26	15,697	2
Swedish Leaf	128	20	35	2
Synthetic Control	60	7	9	1
Two Patterns	128	22	31	2
U Wave Gesture Library X	315	12	418	2
U Wave Gesture Library Y	315	6	382	1
U Wave Gesture Library Z	315	10	393	2
Wafer	152	17	55	2
Yoga	426	26	945	2

Table 4: Time-series length and the number of features after LDG dimensionality reduction as well as a comparison of the testing time, in milliseconds, required to perform reliable local QDA classification with the näive Bayes quadratic constraint set and jointly Gaussian estimation. The test time shown measures the average time, per test sample, to perform reliable classification at time $t = 1$. Therefore, this is a worst case test time in terms of real-time performance as the number of unknowns in the optimization problem for reliable classification is maximized at time $t = 1$.

7.5 Comparison to Other Methods

In this section, we compare the performance of our reliable incomplete data classifier to ECTS (Xing et al., 1998) and several baselines. For all experiments in this section, we use the näive Bayes quadratic constraint set because it proved to be uniformly better than the box constraint set across all experiments in Section 7.2, while not being as overly conservative as the Chebyshev set. We also use the jointly Gaussian estimation method as it is faster to compute than the GMM method, particularly for the long time-series with many classes (the three U Wave Gesture Library data sets and two Non-Invasive Fetal ECG data sets). We also only show results for local QDA, as the reliable local QDA classifier classified earlier than reliable SVM in all experiments of Sections 7.2 and 7.3.

ECTS trades off between the objectives of classifying early and ensuring that early labels meet final labels by using a parameter that varies between zero and one, with zero resulting in the earliest classification time. However, we emphasize that this parameter is not the same as our reliability parameter τ, in that it provides no guarantee on reliability of the early predictions, but is instead a knob that the user can tune to trade off between earliness and reliability. Xing et al. (1998) set this parameter to 0 in the majority of their experiments. We compare to ECTS by varying this parameter MS $\in \{0, 0.05, 0.1, 0.2, 0.4, 0.8\}$.

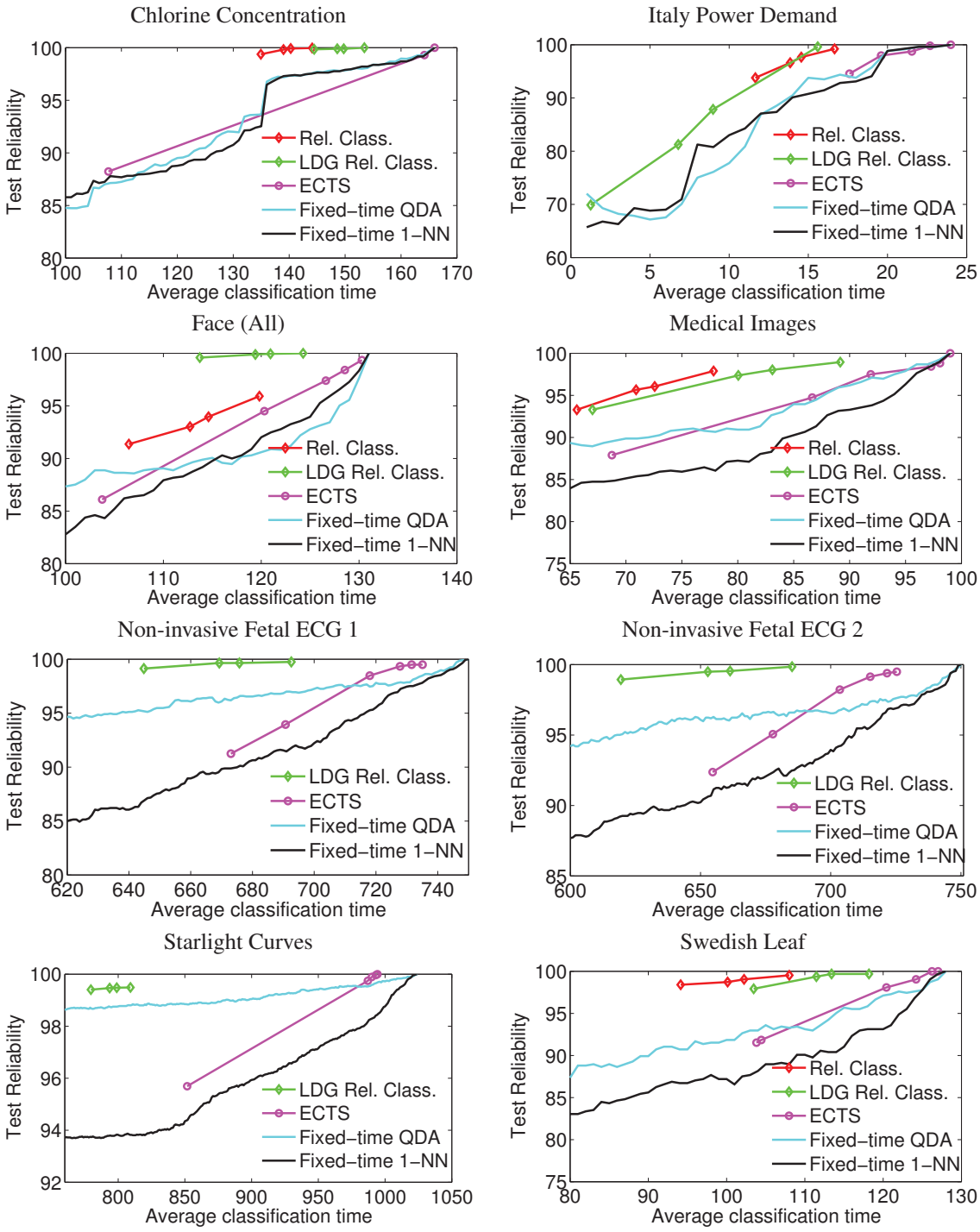


Figure 8: Average classification time vs test reliability for reliable incomplete local QDA classification (Rel. Class.), reliable incomplete local QDA classification with LDG features (LDG Rel. Class.), ECTS, Fixed-time local QDA, and Fixed-time 1-NN.

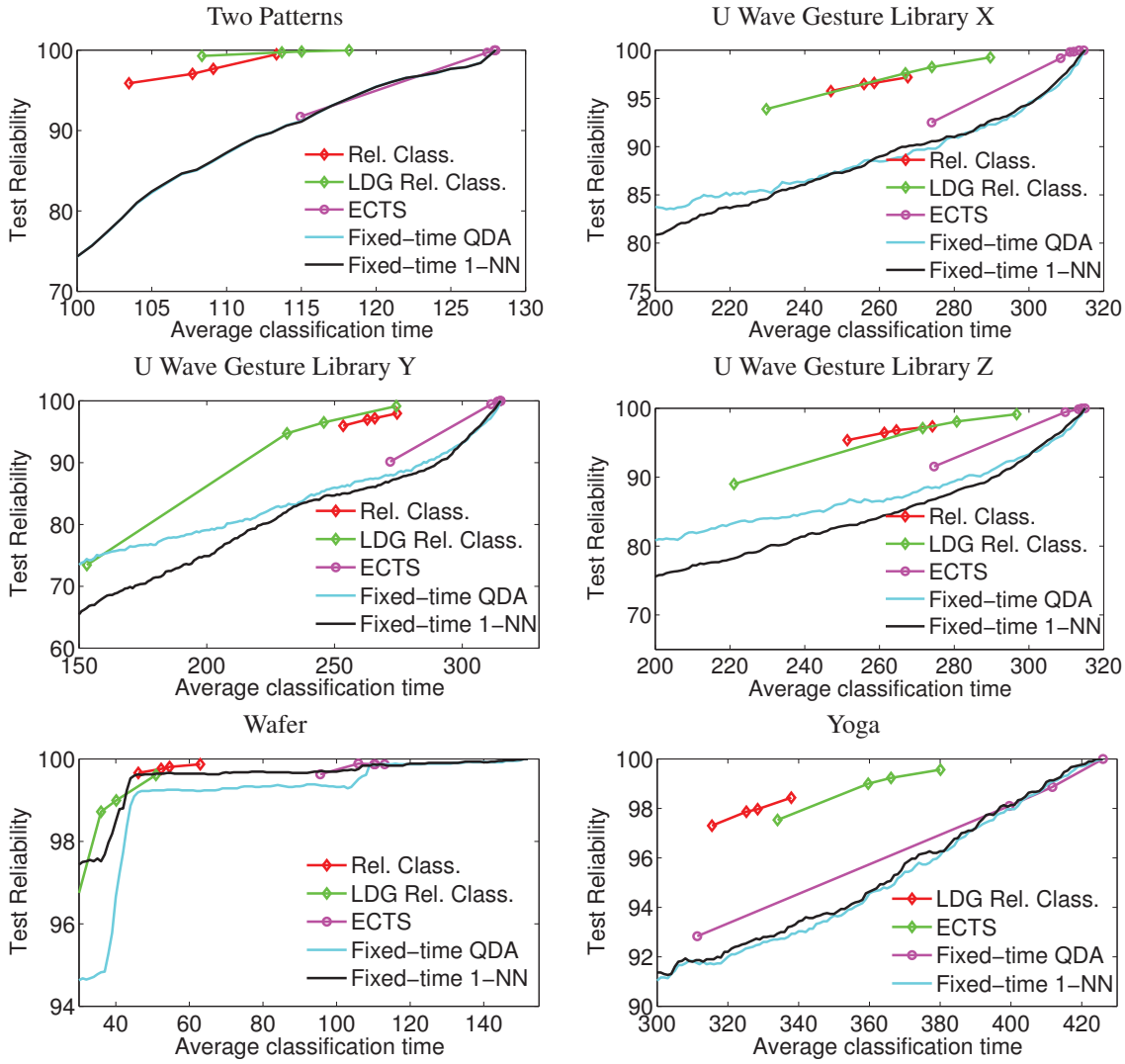


Figure 9: Average classification time vs test reliability for reliable incomplete local QDA classification (Rel. Class.), reliable incomplete local QDA classification with LDG features (LDG Rel. Class.), ECTS, Fixed-time local QDA, and Fixed-time 1-NN.

We also compare to the performance of two baseline methods that we call *Fixed-time local QDA* and *Fixed-time 1-NN*. These methods use no predictive power, but instead classify all test signals at some user specified time: t samples.

The reliability results are shown in Figures 8 and 9. Reliable incomplete local QDA classification and reliable incomplete local QDA classification with LDG features perform well across all experiments. The only times that these methods do not dominate all other methods are when $\tau = 0.001$ on the Italy Power Demand, U Wave Gesture Library Y, and Wafer data sets. For the Starlight Curves and Non-invasive Fetal ECG 1 and 2 data sets, the result of reliable local QDA classification using the raw time-series samples as the features is not shown due to the excessive

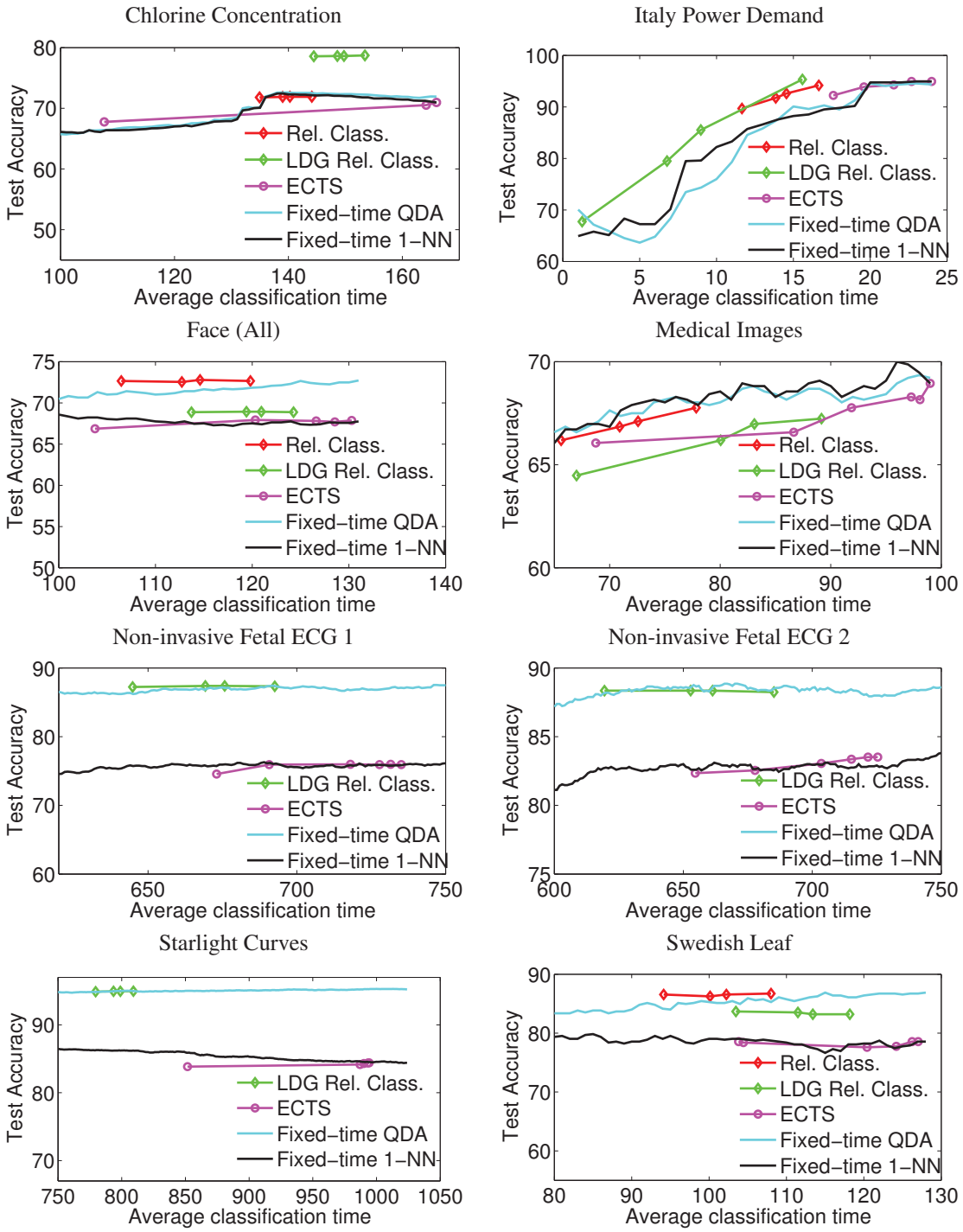


Figure 10: Average classification time vs test accuracy for reliable incomplete local QDA classification (Rel. Class.), reliable incomplete local QDA classification with LDG features (LDG Rel. Class.), ECTS, Fixed-time local QDA, and Fixed-time 1-NN.

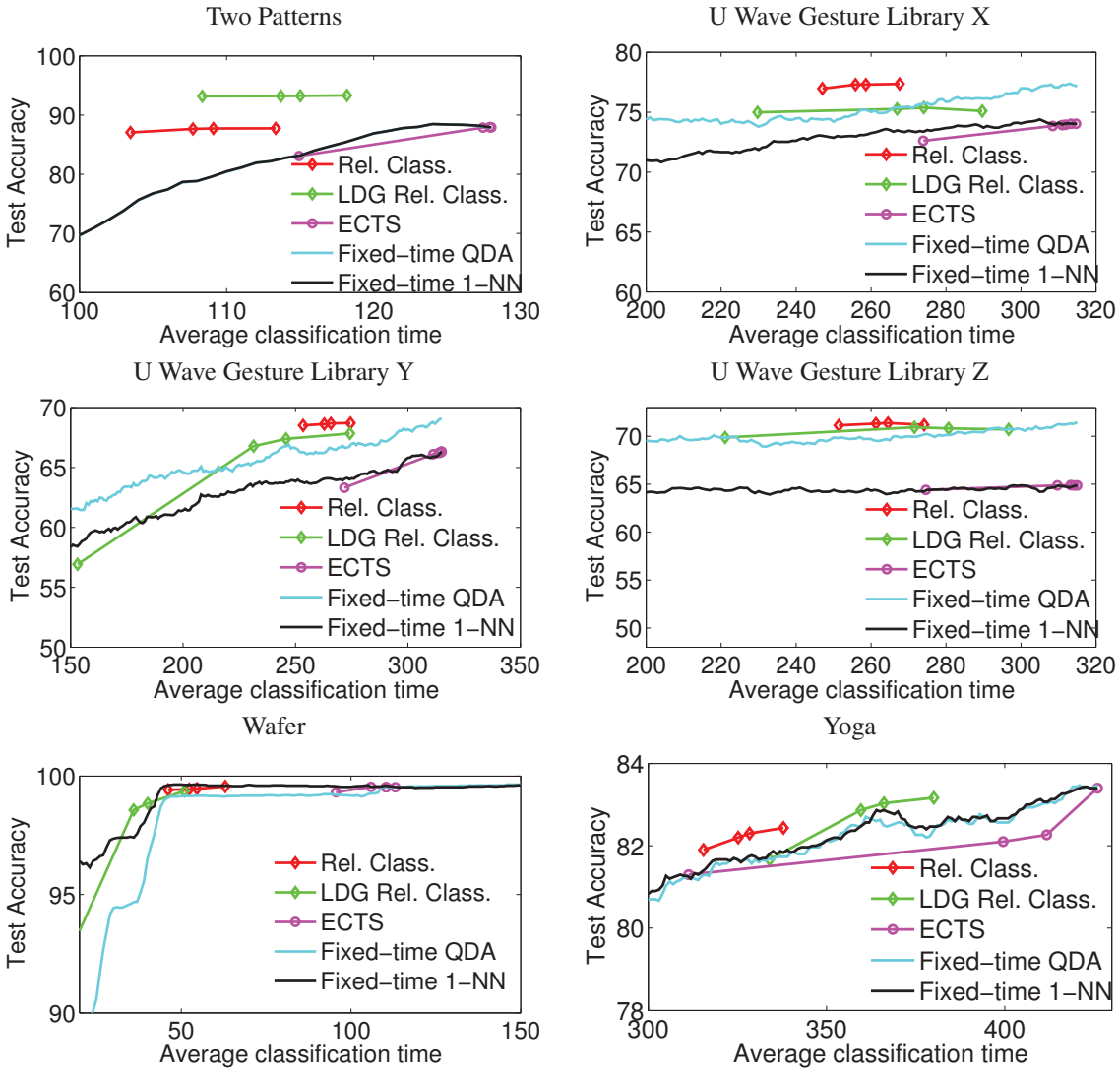


Figure 11: Average classification time vs test accuracy for reliable incomplete local QDA classification (Rel. Class.), reliable incomplete local QDA classification with LDG features (LDG Rel. Class.), ECTS, Fixed-time local QDA, and Fixed-time 1-NN.

run time. However, reliable classification with LDG features performs well on these data sets and is fast, as shown in Table 4.

We also note that the leftmost pink circle in these plots is the earliest possible average classification time that ECTS can achieve, as MS $= 0$ is the smallest possible value for the minimum support parameter. On the other hand, reliable early classification can achieve earlier times than those shown in the figures by setting $\tau < 0.001$ (in fact, setting $\tau = 0$ would result in classifying every signal at time one). Therefore, if someone wanted to set τ by cross-validation on the training data set, the reliable incomplete classifier offers more flexibility than ECTS.

Finally, Figures 10 and 11 plot the test accuracy of the various approaches. The accuracy plots show that local QDA achieves higher accuracy than 1-NN on most of the data sets; therefore, ECTS suffers in comparison to reliable local QDA due to the fact that it attempts to match a less accurate classifier.

The accuracy plots also show that although ECTS is typically more reliable than fixed-time 1-NN, it is less accurate for at least one value of MS on twelve of the fourteen data sets. On the other hand, reliable local QDA using the time-series samples as features is less accurate than fixed-time local QDA on only the Medical Images and Chlorine Concentration data sets. However, on the Chlorine Concentration data sets, reliable local QDA with LDG features greatly exceeds the accuracy of fixed-time local QDA. Furthermore, although it is not shown in the figure, reliable local QDA classification using GMM based estimation exceeds the accuracy of fixed-time local QDA on the Medical Images data set. In fact, the proposed reliable classification approach can be used with a wide variety of features, classifiers, and estimation methods in order to maximize accuracy for a particular application.

8. Discussion and Some Open Questions

We have proposed a practical incomplete decision rule that is a conservative approximation of the optimal rule. Experiments on a set of time-series data showed consistently earlier and more reliable predictions on average than other approaches. We showed that for linear or quadratic classifiers the proposed decision rule can be checked either with an analytic solution or using convex optimization. We only touched on applying the proposed rule to nearest neighbor classifiers, and it is an open question how to apply this approach efficiently to other classification strategies. In particular, we suspect the proposed approach could also be implemented efficiently with decision trees that use a cascade of linear discriminants.

This paper has focused on answering the question "With probability τ, will the classification decision from this incomplete data be the same as from the complete data?" The presented tools can also be used to answer the related question: "If we classify based on the current incomplete data, what is the probability that assigned label will match that which would be chosen using the complete data?" The answer can be computed by finding the largest τ that makes the first question a "Yes," which may require guessing a τ, solving the first question, refining τ up or down depending on the answer, and iterating.

Another related question that can be answered is, "Can we reliably classify as class g with this incomplete data?" That is, there may be only one class (or a subset of classes) which we would like to identify with incomplete data. For example, in determining if a cyst is cancerous or benign, doctors will often have a patient come back every few months to see how it changes over time. There is generally no rush to call it benign, but one would like to classify it as cancerous as soon as that is a reliable class label. This question can be answered by applying the incomplete decision rule given in (2) only to the class of interest.

Acknowledgments

This work was supported by a United States PECASE Award managed by the United States Office of Naval Research, and by the Sandia National Laboratories. Sandia National Laboratories is

a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy National Nuclear Security Administration under contract DE-AC04-94AL85000. We thank Bela A. Frigyik for helpful discussions.

Appendix A. Proofs

Proof of Proposition 1: For the minimum problem (9), the Lagrange dual function is $g(\lambda) = \beta^T m - \frac{1}{4\lambda} B^T R B + b - \lambda\delta$, a concave function of λ, and $g(\lambda)$ is maximized for $\lambda^* = \sqrt{\frac{1}{4\delta} B^T R B}$. Since $\lambda^* \geq 0$, it is dual feasible. Since the objective function is convex, strong duality holds, and thus the maximum of the dual problem equals the minimum of the primal problem. A similar analysis can be performed for the maximum problem.

Proof of Proposition 2: First, note that each pairwise check reduces the number of classes labelled `candidate` by either two classes if the classes tie, or by one class (the loser) if one class dominates. Second, once a class has tied with another class or has been dominated, it cannot be the correct dominating class. Thus the proposed decision process eventually reduces the number of classes labelled `candidate` to either zero or one. If there are zero classes left labelled `candidate`, then all classes have either tied or been dominated, and the above process correctly chooses not to classify. If there is one class remaining that is labelled `candidate` it must be compared to all the classes that tied on their first comparison. It is not necessary to also compare to the classes labelled `dominated` by the transitivity of the domination rule.

Proof of Proposition 3: We first note that in the *Compare* step, the pairwise comparison between classes c and h requires a single minimum computation if c dominates h, and two computations if the classes tie or if h dominates c. Furthermore, we define T to be the number of pairwise comparisons that result in ties during the *Compare* step, and D as the number of pairwise comparisons that do not result in a tie in the *Compare* step (such evaluations necessarily result in one class that was labelled `candidate` being re-labelled `dominated`). Thus, the compare step requires at most $2T + 2D$ minimum calculations.

On the other hand, each pairwise comparison in the *Final Comparison* check requires only a single minimum computation.

There are two cases to consider

Case 1: Consider the case that the *Compare* step in the decision process results in one class left labelled `candidate`. Immediately prior to the *Final Comparison* step, there are $G-1$ classes that have been re-labelled `tied` or `dominated`, and since each tie results in two classes being re-labelled `tied`, it must be that

$$G - 1 = D + 2T. \tag{19}$$

In the *Final Comparison* step, the Gth class must be compared to at most the $2T$ classes labelled `tied`, each of which requires one minimum calculation. Thus the maximum number of calculations needed is

$$2T + 2D + 2T = 2(G-1) \text{ by (19).}$$

Conversely, the best case is that there are no ties, and that each pairwise check requires only a single minimum calculation. This case requires $G-1$ minimum calculations.

Case 2: The second case is that at the end of the *Compare* step there are zero classes labelled `candidate`. Therefore,

$$G = D + 2T, \tag{20}$$

and the total number of comparisons required is

$$2T + 2D = G + D \text{ by (20).}$$

There can be at most $G-2$ comparisons that result in one class dominating the other (otherwise, one class would remain labelled `candidate` after the *Compare* step), so the maximum number of minimum calculations is again $2(G-1)$.

Since it requires at least one minimum calculation change a class label from `candidate` to `dominated` or `tied`, the minimum number of calculations is G.

Appendix B. Gradient Descent Solution for the Quadratic Min and Max Problems

The min problem with quadratic $f(x)$ subject to a quadratic constraint set is written as

$$\begin{aligned} \min_{x \in A} f(x) = &\min_{x} (x-v)^T V (x-v) + b \\ &\text{s.t. } (x-m) R^{-1} (x-m) \leq \delta, \end{aligned} \tag{21}$$

where V is indefinite and R is positive semi-definite. We propose to solve this problem using the two-step gradient descent approach described in Tao and An (1997).

We first reformulate (21) as the *trust region subproblem* (TRSP). Define

$$\begin{aligned} z &= R^{\frac{-1}{2}} (x-m), \\ A &= 2R^{\frac{1}{2}} V R^{\frac{1}{2}}, \\ y &= 2R^{\frac{1}{2}} V (m-v), \\ b_{tsrp} &= b + v^T V v + m^T V m - 2m^T V v. \end{aligned}$$

Then rewrite (21) as

$$\begin{aligned} \min_{x \in A} f(x) = &\min_{z} \frac{1}{2} z^T A z + y^T z + b_{tsrp} \\ &\text{s.t. } \| z \| \leq \sqrt{\delta}. \end{aligned} \tag{22}$$

Let ρ equal the largest eigenvalue of A. The following two-step iteration converges to a z^* that is a local minimum of the TRSP (22):

$$\begin{aligned} \text{Step 1}: &\quad z_{k+1} = z_k - \frac{1}{\rho}(A z_k + y), \\ \text{Step 2}: &\quad z_{k+1} = \min \left[z_{k+1}, \frac{\| z_{k+1} \|}{\sqrt{\delta}} z_{k+1} \right], \end{aligned}$$

where Step 1 computes a gradient step, and Step 2 projects the z_{k+1} found in Step 1 onto the constraint set in (22).

The TRSP has been shown to have at most one local minimum that is not also the global minimum (Martinez, 1994), and thus the above algorithm has proven to be robust in finding the minimum of (22).

Furthermore, for the incomplete data decision rule (7), it is not necessary to find the true minimum over A of $f(x)$, but it is instead sufficient to know only whether or not it is less than or equal to zero. Therefore, the iteration can be stopped early if $z_k^T A z_k + y^T z_k + b_{tsrp} \leq 0$.

Appendix C. Derivation of the Variance for GMM Based Estimation

Let $m_g = \mathrm{E}[X \mid g,z]$, $R_g = \mathrm{COV}[X \mid g,z]$, and $p(g|z)$ be defined as in Section 5.2.

$$
\begin{aligned}
R &= \int_x (x-m)(x-m)^T \; p(x|z)\, dx \\
&= \int_x \sum_{g\in\mathcal{G}} (x-m)(x-m)^T \; p(x,g|z)\, dx \\
&= \sum_{g\in\mathcal{G}} p(g|z) \int_x (x-m)(x-m)^T \; p(x|g,z)\, dx \\
&= \sum_{g\in\mathcal{G}} p(g|z) \int_x \left(xx^T - 2x \sum_{h\in\mathcal{G}} m_h^T p(h|z) + \sum_{q\in\mathcal{G}} \sum_{h\in\mathcal{G}} m_q m_h^T p(q|z) p(h|z) \right) \; p(x|g,z)\, dx \\
&= \sum_{g\in\mathcal{G}} p(g|z) \left(\int_x xx^T - 2xm_g^T + m_g m_g^T \; p(x|g,z)\, dx + \int_x 2xm_g^T - 2x \sum_{h\in\mathcal{G}} m_h^T p(h|z) \; p(x|g,z)\, dx \right. \\
&\qquad \left. - m_g m_g^T + \sum_{q\in\mathcal{G}} \sum_{h\in\mathcal{G}} m_q m_h^T p(q|z) p(h|z) \right) \\
&= \sum_{g\in\mathcal{G}} p(g|z) \left(R_g + 2m_g m_g^T - 2m_g \sum_{h\in\mathcal{G}} m_h^T p(h|z) - m_g m_g^T + \sum_{q\in\mathcal{G}} \sum_{h\in\mathcal{G}} m_q m_h^T p(q|z) p(h|z) \right) \\
&= \sum_{g\in\mathcal{G}} p(g|z) \left(R_g + m_g m_g^T - 2m_g \sum_{h\in\mathcal{G}} m_h^T p(h|z) \right) + \sum_{q\in\mathcal{G}} \sum_{h\in\mathcal{G}} m_q m_h^T p(q|z) p(h|z) \\
&= \sum_{g\in\mathcal{G}} p(g|z) \left(R_g + m_g m_g^T \right) - \sum_{q\in\mathcal{G}} \sum_{h\in\mathcal{G}} m_q m_h^T p(q|z) p(h|z).
\end{aligned}
$$

References

H. S. Anderson, N. Parrish, K. Tsukida, and M. R. Gupta. Reliable early classification of time series. In *ICASSP*, 2012.

P. Armitage. Sequential Analysis with More than Two Alternative Hypotheses, and its Relation to Discriminant Function Analysis. *Journal of the Royal Statistical Society. Series B (Methodological)*, 12(1):137–144, January 1950.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2008.

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

M. Cooke, P. Green, L. Josifovski, and A. Vizinho. Robust automatic speech recognition with missing and unreliable acoustic data. *Speech Communication*, 34(3):267 – 285, 2001.

M. C. Costanza and A. A. Afifi. Comparison of Stopping Rules in Forward Stepwise Discriminant Analysis. *Journal of the American Statistical Association*, 74(368):777–785, January 1979.

K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *Journal Machine Learning Research*, 3:951–991, 2003.

M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. *Intl. Conf. Machine Learning (ICML)*, 2008.

T. Ferguson. *Optimal Stopping Rules and Applications*. E-book available on the author's website., 2001.

J. Friedman, R. Kohavi, and Y. Yun. Lazy decision trees. In *Proc. AAAI*, 1996.

W. J. Fu, E. R. Dougherty, B. Mallick, and R. J. Carroll. How many samples are needed to build a classifier: a general sequential approach. *Bioinformatics*, 21(1):63–70, January 2005.

E. K. Garcia, S. Feldman, M. R. Gupta, and S. Srivastava. Completely lazy learning. *IEEE Trans. Knowledge and Data Engineering*, 22(9):1274–1285, Sept. 2010.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, New York, 2001.

E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana. The UCR time series classification and clustering webpage. 2006.

J. M. Martinez. Local minimizers of quadratic functions on Euclidean balls and spheres. *SIAM J. Optimization*, 4(1):159 – 176, 1994.

S. Pang, S. Ozawa, and N. Kasabov. Incremental linear discriminant analysis for classification of data streams. *IEEE Trans. Systems, Man, and Cybernetics*, 35(5):905–914, October 2005.

N. Parrish and M. R. Gupta. Dimensionality reduction by local discriminative Gaussians. In *Proc. Intl. Conf. on Machine Learning (ICML)*, 2012.

B. Raj and R.M. Stern. Missing-feature approaches in speech recognition. *Signal Processing Magazine, IEEE*, 22(5):101 – 116, 2005.

B. Raj, M. L. Seltzer, and R. M. Stern. Reconstruction of missing features for robust speech recognition. *Speech Communication*, 43(4):275 – 296, 2004.

J. J. Rodriguez and C. J. Alonso. Boosting interval-based literals: Variable length and early classification. In *ECAI'02 Workshop on Knowledge Discovery from (Spatio-) Temporal Data*, 2002.

A. Rogier, T. Donders, Geert J. M. G. van der Heijden, T. Stijnen, and K. G. M. Moons. Review: A gentle introduction to imputation of missing values. *J. of Clinical Epidemiology*, 59(10):1087–1091, 2006.

M. Saar-Tsechansky and F. Provost. Handling missing values when applying classification models. *Journal Machine Learning Research*, 2007.

J. L. Schafer and J. W. Graham. Missing data: Our view of the state of the art. *Psychological Methods*, 7(2):147–177, 2002.

D. Schuurmans and R. Greiner. Learning to classify incomplete examples. In *Computational Learning Theory and Natural Learning Systems: Making Learning Systems Practical*, 2007.

P. D. Tao and L. T. H. An. Convex analysis approach to D.C. programming: theory, algorithms, and applications. *ACTA Mathematica Vietnamica*, 22(1):289 – 355, 1997.

A. Wald. *Sequential Analysis*. John Wiley, 1947.

Z. Xing, J. Pei, and P. S. Yu. Early prediction on time series: a nearest neighbor approach. In *IJCAI*, pages 1297–1302, 1998.

Journal of Machine Learning Research 14 (2013) 3591-3618 Submitted 4/13; Revised 9/13; Published 12/13

Classifier Selection using the Predicate Depth

Ran Gilad-Bachrach RANG@MICROSOFT.COM
Christopher J.C. Burges CBURGES@MICROSOFT.COM
Microsoft Research
1 Microsoft Way
Redmond, WA, 98052
USA

Editor: John Shawe-Taylor

Abstract

Typically, one approaches a supervised machine learning problem by writing down an objective function and finding a hypothesis that minimizes it. This is equivalent to finding the Maximum A Posteriori (MAP) hypothesis for a Boltzmann distribution. However, MAP is not a robust statistic. We present an alternative approach by defining a median of the distribution, which we show is both more robust, and has good generalization guarantees. We present algorithms to approximate this median.

One contribution of this work is an efficient method for approximating the Tukey median. The Tukey median, which is often used for data visualization and outlier detection, is a special case of the family of medians we define: however, computing it exactly is exponentially slow in the dimension. Our algorithm approximates such medians in polynomial time while making weaker assumptions than those required by previous work.

Keywords: classification, estimation, median, Tukey depth

1. Introduction

According to the PAC-Bayesian point of view, learning can be split into three phases. First, a prior belief is introduced. Then, observations are used to transform the prior belief into a posterior belief. Finally, a hypothesis is selected. In this study, we concentrate on the last step. This allows us to propose methods that are independent of the first two phases. For example, the observations used to form the posterior belief can be supervised, unsupervised, semi-supervised, or something entirely different. The most commonly used method for selecting a hypothesis is to select the maximum a posteriori (MAP) hypothesis. For example, many learning algorithms use the following evaluation function (energy function):

$$E(f) = \sum_{i=1}^{n} l(f(x_i), y_i) + r(f) \ , \tag{1}$$

where l is a convex loss function, $\{(x_i, y_i)\}_{i=1}^{n}$ are the observations and r is a convex regularization term. This can be viewed as a prior P over the hypothesis class with density $p(f) = \frac{1}{Z_p} e^{-r(f)}$ and a posterior belief Q with density $q(f) = \frac{1}{Z_q} e^{-E[f]}$. The common practice is then to select the hypothesis that minimizes the evaluation function, that is, the MAP hypothesis. However, this choice has two significant drawbacks. First, since it considers only the maximal point, it misses much of the information encoded in the posterior belief. As a result it is straightforward to construct patho-

logical examples: in Section 2.4 we give an example where the MAP classifier solution disagrees with the Bayes optimal hypothesis on every point, and where the Bayes optimal hypothesis[1] in fact *minimizes* the posterior probability. Second, the MAP framework is sensitive to perturbations in the posterior belief. That is, if we think of the MAP hypothesis as a statistic of the posterior, it has a low breakdown point (Hampel, 1971): in fact, its breakdown point is zero as demonstrated in Section 2.2.

This motivates us to study the problem of selecting the best hypothesis, given the posterior belief. The goal is to select a hypothesis that will generalize well. Two well known methods for achieving this are the Bayes optimal hypothesis, and the Gibbs hypothesis, which selects a random classifier according to the posterior belief. However the Gibbs hypothesis is non-deterministic, and in most cases the Bayes optimal hypothesis is not a member of the hypothesis class; these drawbacks are often shared by other hypothesis selection methods. This restricts the usability of these approaches. For example, in some cases, due to practical constraints, only a hypothesis from a given class can be used; ensembles can be slow and can require large memory footprint. Furthermore stochasticity in the predictive model can make the results non-reproducible, which is unacceptable in many applications, and even when acceptable, makes the application harder to debug. Therefore, in this work we limit the discussion to the following question: given a hypothesis class $\mathcal{F}$ distributed according to a posterior belief Q, how can one select a hypothesis $f \in \mathcal{F}$ that will generalize well? We further limit the discussion to the binary classification setting.

To answer this question we extend the notions of depth and the multivariate median, that are commonly used in multivariate statistics (Liu et al., 1999), to the classification setting. The depth function measures the centrality of a point in a sample or a distribution. For example, if Q is a probability measure over $\mathbb{R}^d$, the Tukey depth for a point $x \in \mathbb{R}^d$, also known as the half-space depth (Tukey, 1975), is defined as

$$\text{TukeyDepth}_Q(x|Q) = \inf_{H \text{ s.t. } x \in H \text{ and } H \text{ is a halfspace}} Q(H) \ . \tag{2}$$

That is, the depth of a point x is the minimal measure of a half-space that contains it.[2] The Tukey depth also has a minimum entropy interpretation: each hyperplane containing x defines a Bernoulli distribution by splitting the distribution Q in two. Choose that hyperplane whose corresponding Bernoulli distribution has minimum entropy. The Tukey depth is then the probability mass of the halfspace on the side of the hyperplane with the lowest such mass.

The depth function is thus a measure of centrality. The median is then simply defined as the deepest point. It is easy to verify that in the univariate case, the Tukey median is indeed the standard median. In this work we extend Tukey's definition beyond half spaces and define a depth for any hypothesis class which we call the predicate depth. We show that the generalization error of a hypothesis is inversely proportional to its predicate depth. Hence, the median predicate hypothesis, or *predicate median,* has the best generalization guarantee. We present algorithms for approximating the predicate depth and the predicate median. Since the Tukey depth is a special case of the predicate depth, our algorithms provide polynomial approximations to the Tukey depth and the Tukey median as well. We analyze the stability of the predicate median and also discuss the case where a convex evaluation function $E(f)$ (see Equation (1)) is used to form the posterior belief. We show that in

1. The Bayes optimal hypothesis is also known as the Bayes optimal classifier. It performs a weighted majority vote on each prediction according to the posterior.
2. Note that we can restrict the half spaces in (2) to those half spaces for which x lies on the boundary.

Symbol	Description
$\mathcal{X}$	a sample space
x	an instance $x \in \mathcal{X}$
μ	a probability measure over $\mathcal{X}$
S	a sample of instances, $S = \{x_1, \ldots, x_u\}$.
$\mathcal{F}$	a function class. $f \in \mathcal{F}$ is a function $f : \mathcal{X} \mapsto \pm 1$.
f, g	functions in the function class $\mathcal{F}$
P, Q, Q'	probability measures over $\mathcal{F}$
T	a sample of functions, $T = \{f_1, \ldots, f_n\}$.
$D_Q(f \mid x)$	The depth of the function f on the instance x with respect to the measure Q.
$D_Q(f)$	The depth of the function f with respect to the measure Q.
$D_Q^{\delta,\mu}(f)$	The δ-insensitive depth of f with respect to Q and μ.
$\hat{D}_T(f \mid x)$	The *empirical depth* of f on the instance x with respect to the sample T
$\hat{D}_T^S(f)$	The *empirical depth* of f with respect to the samples T and S.
ν	A probability measure over $\mathcal{X} \times \{\pm 1\}$
$\mathcal{S}$	a sample $\{(x_i, y_i)\}_{i=1}^m$ from $(\mathcal{X} \times \{\pm 1\})^m$
$R_\nu(f)$	The generalization error of f: $R_\nu(f) = \Pr_{(x,y)\sim\nu}[f(x) \neq y]$.
$R_\mathcal{S}(f)$	The empirical error of f: $R_\mathcal{S}(f) = \Pr_{(x,y)\sim\mathcal{S}}[f(x) \neq y]$.

Table 1: A summary of the notation used in this work

this special case, the average hypothesis has a depth of at least $1/e$, independent of the dimension. Hence, it enjoys good generalization bounds.

In the first part of this work we introduce the notion of predicate depth. We discuss its properties and contrast them with the properties of the MAP estimator. In the second part we discuss algorithmic aspects. We address both the issues of approximating depth and of approximating the deepest hypothesis, that is, the predicate median. Table 1 contains a summery of the notation we use.

2. The Predicate Depth: Definitions and Properties

In this study, unlike Tukey who used the depth function on the instance space, we view the depth function as operating on the dual space, that is the space of classification functions. Moreover, the definition here extends beyond the linear case to any function class. The depth function measures the agreement of the function f with the weighted majority vote on x. A deep function is a function that will always have a large agreement with its prediction among the class $\mathcal{F}$.

Definition 1 *Let $\mathcal{F}$ be a function class and let Q be a probability measure over $\mathcal{F}$. The predicate depth of f on the instance $x \in \mathcal{X}$ with respect to Q is defined as*

$$D_Q(f \mid x) = \Pr_{g\sim Q}[g(x) = f(x)] \ .$$

The predicate depth of f with respect to Q is defined as

$$D_Q(f) = \inf_{x\in\mathcal{X}} D_Q(f \mid x) = \inf_{x\in\mathcal{X}} \Pr_{g\sim Q}[g(x) = f(x)] \ .$$

The Tukey-Depth is a special case of this definition as discussed in section 2.1. We can now define the predicate median:

Definition 2 *Let $\mathcal{F}$ be a function class and let Q be a probability measure over $\mathcal{F}$. f^* is a predicate median of $\mathcal{F}$ with respect to Q if*

$$\forall f \in \mathcal{F},\ D_Q(f) \leq D_Q(f^*)\ .$$

We show later, in Theorem 13, that if $\mathcal{F}$ is closed then the median always exists, for every probability measure Q. The depth $D_Q(f)$ is defined as the infimum over all points $x \in \mathcal{X}$. However, for our applications, we can tolerate some instances $x \in \mathcal{X}$ which have small depth, as long as most of the instances have large depth. Therefore, we define the δ-insensitive depth:

Definition 3 *Let $\mathcal{F}$ be a function class and let Q be a probability measure over $\mathcal{F}$. Let μ be a probability measure over $\mathcal{X}$ and let $\delta \geq 0$. The δ-insensitive depth of f with respect to Q and μ is defined as*

$$D_Q^{\delta,\mu}(f) = \sup_{X' \subseteq \mathcal{X}, \mu(X') \leq \delta} \inf_{x \in \mathcal{X} \setminus X'} D_Q(f\,|\,x)\ .$$

The δ-insensitive depth function relaxes the infimum in the depth definition. Instead of requiring that the function f always have a large agreement in the class $\mathcal{F}$, the δ-insensitive depth makes this requirement on all but a set of the instances with probability mass δ.

With these definitions in hand, we next provide generalization bounds for deep hypotheses. The first theorem shows that the error of a deep function is close to the error of the Gibbs classifier.

Theorem 4 ***Deep vs. Gibbs***

Let Q be a measure on $\mathcal{F}$. Let ν be a measure on $\mathcal{X} \times \{\pm 1\}$ with the marginal μ on $\mathcal{X}$. For every f the following holds:

$$R_\nu(f) \leq \frac{1}{D_Q(f)} E_{g \sim Q}[R_\nu(g)] \tag{3}$$

and

$$R_\nu(f) \leq \frac{1}{D_Q^{\delta,\mu}(f)} E_{g \sim Q}[R_\nu(g)] + \delta\ . \tag{4}$$

Note that the term $E_{g \sim Q}[R_\nu(g)]$ is the expected error of the Gibbs classifier (which is not necessarily the same as the expected error of the Bayes optimal hypothesis). Hence, this theorem proves that the generalization error of a deep hypothesis cannot be large, provided that the expected error of the Gibbs classifier is not large.

Proof For every $x^* \in \mathcal{X}$ we have that

$$\begin{aligned}
& \Pr_{g \sim Q, (x,y) \sim \nu}[g(x) \neq y\,|\,x = x^*] \\
\geq\ & \Pr_{g \sim Q, (x,y) \sim \nu}[f(x) \neq y \text{ and } g(x) = f(x)\,|\,x = x^*] \\
=\ & \Pr_{(x,y) \sim \nu}[f(x) \neq y | x = x^*] \Pr_{g \sim Q, (x,y) \sim \nu}[g(x) = f(x)\,|\,x = x^* \text{ and } f(x) \neq y] \\
=\ & \Pr_{(x,y) \sim \nu}[f(x) \neq y | x = x^*] \Pr_{g \sim Q, (x,y) \sim \nu}[g(x) = f(x)\,|\,x = x^*] \\
=\ & \Pr_{(x,y) \sim \nu}[f(x) \neq y | x = x^*] D_Q(f\,|\,x^*) \geq \Pr_{(x,y) \sim \nu}[f(x) \neq y | x = x^*] D_Q(f)\ .
\end{aligned}$$

First we prove (4). Define the set $Z = \left\{x : D_Q(f|x) < D_Q^{\delta,\mu}(f)\right\}$. Clearly $\mu(Z) \leq \delta$. By slight abuse of notation, we define the function $Z(x)$ such that $Z(x) = 1$ if $x \in Z$ and $Z(x) = 0$ if $x \notin Z$. With this definition we have

$$\begin{aligned}
\frac{1}{D_Q^{\delta,\mu}(f)} E_{g\sim Q}[R_\nu(g)] + \delta &\geq E_{x^*\sim\mu}\left[\frac{1}{D_Q^{\delta,\mu}(f)} \Pr_{g\sim Q,(x,y)\sim\nu}[g(x) \neq y | x = x^*] + Z(x^*)\right] \\
&\geq E_{x^*\sim\mu}\left[\frac{1}{D_Q^{\delta,\mu}(f)} \Pr_{(x,y)\sim\nu}[f(x) \neq y | x = x^*] D_Q(f|x^*) + Z(x^*)\right] \\
&\geq E_{x^*\sim\mu}\left[\Pr_{(x,y)\sim\nu}[f(x) \neq y | x = x^*]\right] = R_\nu(f) \ .
\end{aligned}$$

(3) follows in the same way by setting both Z and δ to be zero. ■

Theorem Deep vs. Gibbs (Theorem 4) bounds the ratio of the generalization error of the Gibbs classifier and the generalization error of a given classifier as a function of the depth of the given classifier. For example, consider the Bayes optimal classifier. By definition, the depth of this classifier is at least one half; thus Theorem 4 recovers the well-known result that the generalization error of the Bayes optimal classifier is at most twice as large as the generalization error of the Gibbs classifier.

Next, we combine Theorem Deep vs. Gibbs (Theorem 4) with PAC-Bayesian bounds (McAllester, 1999) to bound the difference between the training error and the generalization error. We use the version of the PAC-Bayesian bounds in Theorem 3.1. of Germain et al. (2009).

Theorem 5 ***Generalization Bounds***

Let ν be a probability measure on $\mathcal{X} \times \{\pm 1\}$, let P be a fixed probability measure on $\mathcal{F}$ chosen a priori, and let $\delta, \kappa > 0$. For a proportion $1-\delta$ of samples $S \sim \nu^m$,

$$\forall Q,\ \forall f,\ R_\nu(f) \leq \frac{1}{(1-e^{-\kappa}) D_Q(f)} \left(\kappa E_{g\sim Q}[R_S(g)] + \frac{1}{m}\left[KL(Q||P) + \ln\frac{1}{\delta}\right]\right) \ .$$

Furthermore, for every $\delta' > 0$, for a proportion $1-\delta$ of samples $S \sim \nu^m$,

$$\forall Q,\ \forall f,\ R_\nu(f) \leq \frac{1}{(1-e^{-\kappa}) D_Q^{\delta',\mu}(f)} \left(\kappa E_{g\sim Q}[R_S(g)] + \frac{1}{m}\left[KL(Q||P) + \ln\frac{1}{\delta}\right]\right) + \delta' \ ,$$

where μ is the marginal of ν on $\mathcal{X}$.

Proof Applying the bounds in Theorem 4 to the PAC-Bayesian bounds in Theorem 3.1 of Germain et al. (2009) yields the stated results. ■

The generalization bounds theorem (Theorem 5) shows that if a deep function exists, then it is expected to generalize well, provided that the PAC-Bayes bound for Q is sufficiently smaller than the depth of f. This justifies our pursuit to find the deepest function, that is, the median. However, the question remains: are there any deep functions? In the following section we show that this indeed the case for linear classifiers.

2.1 Depth for Linear Classifiers

In this section we discuss the special case where the hypothesis class consists of linear classifiers. Our goal is to show that deep functions exist and that the Tukey depth is a special case of the predicate depth. To that end we use a variant of linear classifiers called *linear threshold functions*. We denote by $\mathbb{S} = \left\{x \in \mathbb{R}^d \,:\, \|x\| = 1\right\}$ the unit sphere. In this setting $\mathcal{F} = \mathbb{R}^d$ and $\mathcal{X} = \mathbb{S} \times \mathbb{R}$ such that $f \in \mathcal{F}$ operates on $x = (x_v, x_\theta) \in \mathcal{X}$ by $f(x) = \text{sign}(f \cdot x_v - x_\theta)$.[3] One can think of an instance $x \in \mathcal{X}$ as a combination of a direction, denoted by x_v, and an offset x_θ.

Theorem 6 *Let $\mathcal{X} = \mathbb{S} \times \mathbb{R}$ and $\mathcal{F}$ be the class of linear threshold functions over $\mathcal{X}$. Let Q be a probability measure over $\mathcal{F}$ with density function $q(f)$ such that $q(f) = \frac{1}{Z}\exp(-E(f))$ where $E(f)$ is a convex function. Let $f^* = E_{f\sim Q}[f]$. Then $D_Q(f^*) \geq 1/e$.*

Proof From the definition of $q(f)$ it follows that it is log-concave. Borell (1975) proved that Q is log-concave if and only if q is log-concave. Hence, in the setting of the theorem, Q is log concave. The Mean Voter Theorem of Caplin and Nalebuff (1991) shows that if f is the center of gravity of Q then for every x, $D_Q(f\,|\,x) \geq 1/e$ and thus the center of gravity of Q has a depth of at least $1/e$ (e is Euler's number). Note that since $\mathcal{F} = \mathbb{R}^d$ then the center of gravity is in $\mathcal{F}$. ■

Recall that it is common practice in machine learning to use convex energy functions $E(f)$. For example, SVMs (Cortes and Vapnik, 1995) and many other algorithms use energy functions of the form presented in (1) in which both the loss function and the regularization functions are convex, resulting in a convex energy function. Hence, in all these cases, the median, that is the deepest point, has a depth of at least $1/e$.[4] This leads to the following conclusion:

Conclusion 1 *In the setting of Theorem 6, let $f^* = E_{f\sim Q}[f]$. Let ν be a probability measure on $\mathcal{X} \times \{\pm 1\}$, let P be a probability measure of $\mathcal{F}$ and let $\delta, \kappa > 0$. With a probability greater than $1-\delta$ over the sample $\mathcal{S}$ sampled from ν^m:*

$$R_\nu(f^*) \leq \frac{e}{(1-e^{-\kappa})}\left(\kappa E_{g\sim Q}[R_{\mathcal{S}}(g)] + \frac{1}{m}\left[KL(Q||P) + \ln\frac{1}{\delta}\right]\right) .$$

Proof This results follows from Theorem 5 and Theorem 6. ■

We now turn to show that the Tukey depth is a special case of the predicate depth. Again, we will use the class of linear threshold functions. Since $\mathcal{F} = \mathbb{R}^d$ in this case we will treat $f \in \mathcal{F}$ both as a function and as a point. Therefore, a probability measure Q over $\mathcal{F}$ is also considered as a probability measure over $\mathbb{R}^d$. The following shows that for any $f \in \mathcal{F}$, the Tukey-depth of f and the predicate depth of f are the same.

Theorem 7 *If $\mathcal{F}$ is the class of threshold functions then for every $f \in \mathcal{F}$:*

$$D_Q(f) = \mathit{TukeyDepth}_Q(f) .$$

3. We are overloading notation here: f is treated as both a point in $\mathbb{R}^d$and a function $f(x) : \mathbb{S} \times \mathbb{R} \to \pm 1$.
4. Note that optimization in general finds the MAP, which can be very different from (and less robust than) the median (see Section 2.4).

Proof Every closed half-space H in $\mathbb{R}^d$ is uniquely identified by a vector $z_H \in \mathbb{S}$ orthogonal to its hyperplane boundary and an offset θ_H such that

$$H = \{g : g \cdot z_H \geq \theta_H\} \ .$$

In other words, there is a $1-1$ correspondence between half-spaces and points in $\mathbb{S} \times \mathbb{R}$ such that $H \mapsto (z_H, \theta_H)$ and such that

$$g \in H \ \Leftrightarrow \ g((z_H, \theta_H)) \equiv sign(g \cdot z_H - \theta_H) = 1 \ .$$

The Tukey depth of f is the infimum measure of half-spaces that contain f:

$$\begin{aligned} \text{TukeyDepth}_Q(f) = \inf_{H: f \in H} Q(H) &= \inf_{x: f(x)=1} Q\{g : g(x) = 1\} \\ &= \inf_{x: f(x)=1} D_Q(f \,|\, x) \\ &\geq D_Q(f) \ . \end{aligned}$$

Hence, the Tukey depth cannot be larger than the predicate depth.

Next we show that the Tukey depth cannot be smaller than the predicate depth for if the Tukey depth is smaller than the predicate depth then there exists a half space H such that its measure is smaller than the predicate depth. Let $x = (z_H, \theta_H)$. Since $f \in H$ then $f(x) = 1$ and thus

$$D_Q(f) > Q(H) = Q\{g : g(x) = 1\} = D_Q(f \,|\, x) \geq D_Q(f)$$

which is a contradiction. Therefore, the Tukey depth can be neither smaller nor larger than the predicate depth and so the two must be equal. ■

2.2 Breakdown Point

We now turn to discuss another important property of the hypothesis selection mechanism: the breakdown point. Any solution to the hypothesis selection problem may be viewed as a statistic of the posterior Q. An important property of any such statistic is its stability: that is, informally, by how much must Q change in order to produce an arbitrary value of the statistic? This is usually referred to as the breakdown point (Hampel, 1971). We extend the definition given there as follows:

Definition 8 *Let **Est** be a function that maps probability measures to $\mathcal{F}$. For two probability measures Q and Q' let $\delta(Q, Q')$ be the total variation distance:*

$$\delta\left(Q, Q'\right) = \sup\left\{\left|Q(A) - Q'(A)\right| : A \text{ is measurable}\right\} \ .$$

For every function $f \in \mathcal{F}$ let $d(\textbf{Est}, Q, f)$ be the distance to the closest Q' such that $\textbf{Est}(Q') = f$:

$$d(\textbf{Est}, Q, f) = \inf\left\{\delta\left(Q, Q'\right) : \textbf{Est}\left(Q'\right) = f\right\} \ .$$

*The breakdown **Est** at Q is defined to be*

$$\textbf{breakdown}(\textbf{Est}, Q) = \sup_{f \in \mathcal{F}} d(\textbf{Est}, Q, f) \ .$$

This definition may be interpreted as follows; if $s = \textbf{breakdown}(\textbf{Est}, Q)$ then for every $f \in \mathcal{F}$, we can force the estimator **Est** to use f as its estimate by changing Q by at most s in terms of total variation distance. Therefore, the larger the breakdown point of an estimator, the more stable it is with respect to perturbations in Q.

As mentioned before, our definition of the breakdown point for an estimator stems from the work of Hampel (1971) who was the first to introduce the concept. Since then, different modifications have been suggested to address different scenarios. Davies and Gather (2005) discuss many of these definitions. He (2005) noted that one can make the breakdown point trivial, for instance, if **Est** is a fixed estimator that is not affected by its input, it has the best possible breakdown point of 1. Moreover, it suffices to have a single function f that cannot be produced as the output of **Est** to make the above definition trivial. To prevent these pathologies, Definition 8 should only be used when **Est** is such that for every f there exists Q' for which $\textbf{Est}(Q') = f$ which is the case for the estimators we study here.

The following theorem lower bounds the stability of the median estimator as a function of its depth.

Theorem 9 *Let Q be a posterior over $\mathcal{F}$. Let*

$$\begin{aligned} X' &= \{x \in \mathcal{X} \text{ s.t. } \forall f_1, f_2 \in \mathcal{F},\ f_1(x) = f_2(x)\} \text{ and} \\ p &= \inf_{x \in \mathcal{X} \setminus X', y \in \{\pm 1\}} Q\{f : f(x) = y\} \ . \end{aligned}$$

If d is the depth of the median for Q then $\textbf{breakdown}(\textbf{median}, Q) \geq \frac{d-p}{2}$.

Proof Let $\varepsilon > 0$. There exists $\hat{f}$ and $\hat{x}$ such that $Q\left\{f : f(\hat{x}) = \hat{f}(\hat{x})\right\} < p + \varepsilon$. Let f^* be the median of Q. Let Q' be such that $\hat{f}$ is the median of Q', so that

$$D_{Q'}(f^*) \leq D_{Q'}\left(\hat{f}\right) \ .$$

Note that for every f we have that

$$|D_Q(f) - D_{Q'}(f)| \leq \delta\left(Q, Q'\right) \ .$$

This follows since

$$\begin{aligned} |D_Q(f) - D_{Q'}(f)| &= \left|\inf_x \left(Q\left\{f' : f'(x) = f(x)\right\}\right) - \inf_x \left(Q'\left\{f' : f'(x) = f(x)\right\}\right)\right| \\ &\leq \delta\left(Q, Q'\right) \ . \end{aligned}$$

Since $D_Q\left(\hat{f}\right) < p + \varepsilon$ then

$$d - \delta\left(Q, Q'\right) \leq D_{Q'}(f^*) \leq D_{Q'}\left(\hat{f}\right) < p + \varepsilon + \delta\left(Q, Q'\right) \ .$$

Hence

$$\delta\left(Q, Q'\right) > \frac{d - p - \varepsilon}{2}$$

and thus

$$\textbf{breakdown}(\textbf{median}, Q) > \frac{d - p - \varepsilon}{2} \ .$$

Since this is true for every $\varepsilon > 0$ it follows that

$$\textbf{breakdown}(\textbf{median}, Q) \geq \frac{d-p}{2} \ .$$

■

2.3 Geometric Properties of the Depth Function

In this section we study the geometry of the depth function. We show that the level sets of the depth functions are convex. We also show that if the function class $\mathcal{F}$ is closed then the median exists. First, we define the term *convex hull* in the context of function classes:

Definition 10 *Let $\mathcal{F}$ be a function class and let $g, f_1, \ldots, f_n \in \mathcal{F}$. We say that g is in the convex-hull of $f_1, \ldots, f_n$ if for every x there exists $j \in 1, \ldots, n$ such that $g(x) = f_j(x)$.*

Theorem 11 ***Convexity***
Let $\mathcal{F}$ be a function class with a probability measure Q. If g is in the convex-hull of $f_1, \ldots, f_n$ then

$$D_Q(g) \geq \min_j D_Q(f_j) \ .$$

Furthermore, if μ is a measure on $\mathcal{X}$ and $\delta \geq 0$ then

$$D_Q^{\delta,\mu}(g) \geq \min_j D_Q^{\frac{\delta}{n},\mu}(f_j) \ .$$

Proof Let g be a function. If g is in the convex-hull of $f_1, \ldots, f_n$ then for every x there exists j such that $g(x) = f_j(x)$ and hence

$$D_Q(g\,|\,x) = D_Q(f_j\,|\,x) \geq \min_j D_Q(f_j) \ ,$$

thus $D_Q(g) \geq \min_j D_Q(f_j)$. For $\delta > 0$ let

$$\Delta \quad = \quad \left\{x : D_Q(g\,|\,x) \leq D_Q^{\delta,\mu}(g)\right\}$$

and for $j = 1, \ldots, n$

$$\Delta_j = \left\{x \in \Delta : f_j(x) = g(x)\right\} \ .$$

Since g is in the convex-hull of $f_1, \ldots, f_n$ implies that $\cup_j \Delta_j = \Delta$ and therefore

$$\sum_j \mu(\Delta_j) \geq \mu(\Delta) \geq \delta \ .$$

Hence, there exists j such that $\mu(\Delta_j) \geq \delta/n$ which implies that

$$D_Q^{\delta,\mu}(g) \geq D_Q^{\frac{\delta}{n},\mu}(f_j) \geq \min_j D_Q^{\frac{\delta}{n},\mu}(f_j) \ .$$

■

Next we prove the existence of the median when the function class is closed. We begin with the following definition:

Definition 12 *A function class $\mathcal{F}$ is* closed *if for every sequence $f_1, f_2, \ldots \in \mathcal{F}$ there exists $f^* \in \mathcal{F}$ such that for every $x \in \mathcal{X}$, if $\lim_{n\to\infty} f_n(x)$ exists then $f^*(x) = \lim_{n\to\infty} f_n(x)$.*

With this definition in hand we prove the following:

Theorem 13 *If $\mathcal{F}$ is closed then the median of $\mathcal{F}$ exists with respect to any probability measure Q.*

Proof Let $D = \sup_f D_Q(f)$ and let f_n be such that $D_Q(f_n) > D - 1/n$. Let $f^* \in \mathcal{F}$ be the limit of the series $f_1, f_2, \ldots$. We claim that $D_Q(f^*) = D$. Since D is the supermum of the depth values, it is clear that $D_Q(f^*) \leq D$. Note that from the construction of f^* we have that for every $x \in \mathcal{X}$ and every N there exists $n > N$ such that $f^*(x) = f_n(x)$. Therefore, if $D_Q(f^*) < D$ then there exists x such that $D_Q(f^*|x) < D$. Hence, there is a subsequence $n_k \to \infty$ such that $f_{n_k}(x) = f^*(x)$ and thus

$$D_Q(f_{n_k}) \leq D_Q(f_{n_k}|x) = D_Q(f^*|x) < D \ .$$

But this is a contradiction since $\lim_{k\to\infty} D_Q(f_{n_k}) = D$. Hence, for every x, $D_Q(f^*|x) \geq D$ and thus $D_Q(f^*) \geq D$ which completes the proof. ■

2.4 The Maximum A Posteriori Estimator

So far, we have introduced the predicate depth and median and we have analyzed their properties. However, the common solution to the hypothesis selection problem is to choose the maximum a posteriori estimator. In this section we point out some limitations of this approach. We will show that in some cases, the MAP method has poor generalization. We also show that it is very sensitive in the sense that the breakdown point of the MAP estimator is always zero.

2.4.1 LEARNING AND REGULARIZATION

The most commonly used method for selecting a hypothesis is to select the maximum a posteriori (MAP) hypothesis. For example, in Support Vector Machines (Cortes and Vapnik, 1995), one can view the objective function of SVM as proportional to the log-likelihood function of an exponential probability. From this perspective, the regularization term is proportional to the log-likelihood of the prior. SVM, Lasso (Tibshirani, 1994) and other algorithms use the following evaluation function (energy function):

$$E(f) = \sum_{i=1}^{n} l(f(x_i), y_i) + r(f) \ ,$$

where l is a convex loss function, $\{(x_i, y_i)\}_{i=1}^n$ are the observations and r is a convex regularization term. This can be viewed as if there is a prior P over the hypothesis class with a density function

$$p(f) = \frac{1}{Z_p} e^{-r(f)} \ ,$$

and a posterior belief Q with a density function

$$q(f) = \frac{1}{Z_q} e^{-E[f]} \ .$$

The common practice in these cases is to select the hypothesis that minimizes the evaluation function. Hence these methods select the MAP hypothesis.

2.4.2 THE MAP ESTIMATOR CAN GENERALIZE POORLY

Since the MAP estimator looks only at the peak of the distribution it can be very misleading. Here we give an example for which the MAP estimator disagrees with the Bayes optimal hypothesis on every instance while the median hypothesis agrees with the Bayes optimal hypothesis everywhere. Moreover, the Bayes optimal hypothesis happens to be a member of the hypothesis class. Therefore, it is also the predicate median. Hence, in this case, the MAP estimator fails to represent the belief. The rest of this sub-section is devoted to explaining the details of this example.

Assume that the sample space $\mathcal{X}$ is a set of N discrete elements indexed by integers $1,...,N$. To simplify the exposition we will collapse notation and take $\mathcal{X} = \{1,...,N\}$. The function class $\mathcal{F}$ consists of $N+1$ functions defined as follows: for every $i \in \{1,\ldots,N-1\}$ the function f_i is defined to be

$$f_i(x) = \begin{cases} 1 & \text{if } x \equiv i \text{ or } x \equiv i+1 \\ 0 & \text{otherwise} \end{cases} .$$

Additionally, $\mathcal{F}$ contains the constant functions f_- and f_+ that assign the values 0 and 1, respectively, to every input. Furthermore, assume that there is ε random label noise in the system for some $0 < \varepsilon < 1/2$, and that no further information is available. Thus, the prior is the non-informative uniform prior over the $N+1$ functions.

Assume that a training set consisting of just two examples is available, where the examples are $(x_1 = 1, y_1 = 1)$ and $(x_2 = 3, y_2 = 1)$. Given the ε random label noise, the posterior is easily computed as

$$Q\{f_+\} = \frac{(1-\varepsilon)^2}{Z},\ Q(f_-) = \frac{\varepsilon^2}{Z},\ Q\{f_{i=1,2,3}\} = \frac{\varepsilon(1-\varepsilon)}{Z},\ Q\{f_{i>3}\} = \frac{\varepsilon^2}{Z}$$

where Z is the partition function. Note that under this posterior, for every x,

$$\Pr_{f\sim Q}[f(x) = 1] \leq \frac{(1-\varepsilon)^2 + 2\varepsilon(1-\varepsilon)}{Z} = \frac{1-\varepsilon^2}{Z}$$

while

$$\Pr_{f\sim Q}[f(x) = 0] \geq \frac{(N-2)\varepsilon^2}{Z} .$$

Therefore, if $N > 1 + 1/\varepsilon^2$, for any x, the probability that it is assigned class 0 is larger than the probability that it is assigned class 1. Therefore the Bayes classifier is the function f_-. Since the Bayes classifier is in the function class, it is also the predicate median. However, the MAP estimator is the function f_+. Thus in this case the MAP estimator disagrees with the Bayes optimal hypothesis (and the predicate median) on the entire sample space. Note also that the Bayes optimal hypothesis f_0 has the lowest density in the distribution Q. Hence, in this case, the minimizer of the posterior is a better estimator than the maximizer of the posterior.

2.4.3 THE BREAKDOWN POINT OF THE MAP ESTIMATOR

In Definition 8 we defined the breakdown point of an estimator. We showed in Theorem 9 that the breakdown point of the median hypothesis is lower bounded by a function of its depth. We would like to contrast this with the breakdown point of the MAP estimator. We claim that the breakdown point of the MAP estimator is zero, for continuous concept classes.

Algorithm 1 Depth Estimation Algorithm

Inputs:

- A sample $S = \{x_1, \ldots, x_u\}$ such that $x_i \in \mathcal{X}$
- A sample $T = \{f_1, \ldots, f_n\}$ such that $f_j \in \mathcal{F}$
- A function f

Output:

- $\hat{D}_T^S(f)$ - an approximation for the depth of f

Algorithm:

1. for $i = 1, \ldots, u$ compute $\hat{D}_T(f \,|\, x_i) = \frac{1}{n} \sum_j 1_{f_j(x_i) = f(x_i)}$
2. return $\hat{D}_T^S(f) = \min_i \hat{D}(f \,|\, x_i)$

In order for the MAP estimator to be well defined, assume that Q is a Lebesgue measure such that q is the density function of Q and q is bounded by some finite M. Let $f_0 \in \mathcal{F}$ and consider $Q'$with the density function:

$$q'(f) = \begin{cases} M+1 & \text{if } f = f_0 \\ q(f) & \text{otherwise} \end{cases}.$$

While the total variation distance between Q and Q' is zero, the MAP estimator for Q' is f_0. Therefore, for every f_0 we can introduce a zero measure change to Q that will make f_0 the MAP estimator. Hence, the breakdown point of the MAP estimator is zero.

3. Measuring Depth

So far, we have motivated the use of depth as a criterion for selecting a hypothesis. Finding the deepest function, even in the case of linear functions, can be hard but some approximation techniques have been presented (see Section 4.5). In this section we focus on algorithms that measure the depth of functions. The main results are an efficient algorithm for approximating the depth uniformly over the entire function class and an algorithm for approximating the median.

We suggest a straightforward method to measure the depth of a function. The depth estimation algorithm (Algorithm 1) takes as inputs two samples. One sample, $S = \{x_1, \ldots, x_u\}$, is a sample of points from the domain $\mathcal{X}$. The other sample, $T = \{f_1, \ldots, f_n\}$, is a sample of functions from $\mathcal{F}$. Given a function f for which we would like to compute the depth, the algorithm first estimates its depth on the points $x_1, \ldots, x_u$ and then uses the minimal value as an estimate of the global depth. The depth on a point x_i is estimated by counting the fraction of the functions $f_1, \ldots, f_n$ that make the same prediction as f on the point x_i. Since samples are used to estimate depth, we call the value returned by this algorithm, $\hat{D}_T^S(f)$, the empirical depth of f.

Despite its simplicity, the depth estimation algorithm can provide good estimates of the true depth. The following theorem shows that if the x_i's are sampled from the underlying distribution over $\mathcal{X}$, and the f_j's are sampled from Q, then the empirical depth is a good estimator of the true

depth. Moreover, this estimate is uniformly good over all the functions $f \in \mathcal{F}$. This will be an essential building block when we seek to find the median in Section 3.1.

Theorem 14 ***Uniform convergence of depth***

Let Q be a probability measure on $\mathcal{F}$ and let μ be a probability measure on $\mathcal{X}$. Let $\varepsilon, \delta > 0$. For every $f \in \mathcal{F}$ let the function f_δ be such that $f_\delta(x) = 1$ if $D_Q(f|x) \leq D_Q^{\delta,\mu}(f)$ and $f_\delta(x) = -1$ otherwise. Let $\mathcal{F}_\delta = \{f_\delta\}_{f\in\mathcal{F}}$ .Assume $\mathcal{F}_\delta$ has a finite VC dimension $d < \infty$ and define $\phi(d,k) = \sum_{i=0}^{d} \binom{k}{i}$ if $d < k$, $\phi(d,k) = 2^k$ otherwise. If S and T are chosen at random from μ^u and Q^n respectively such that $u \geq 8/\delta$ then with probability

$$1 - u\exp\left(-2n\varepsilon^2\right) - \phi(d, 2u)\, 2^{1-\delta u/2}$$

the following holds:

$$\forall f \in \mathcal{F},\ D_Q(f) - \varepsilon \leq D_Q^{0,\mu}(f) - \varepsilon \leq \hat{D}_T^S(f) \leq D_Q^{\delta,\mu}(f) + \varepsilon$$

where $\hat{D}_T^S(f)$ is the empirical depth computed by the depth measure algorithm.

First we recall the definition of ε-nets of Haussler and Welzl (1986):

Definition 15 *Let μ be a probability measure defined over a domain $\mathcal{X}$. Let R be a collection of subsets of $\mathcal{X}$. An ε-net is a finite subset $A \subseteq \mathcal{X}$ such that for every $r \in R$, if $\mu(r) \geq \varepsilon$ then $A \cap r \neq \emptyset$.*

The following theorem shows that a random set of points forms an ε-net with high probability if the VC dimension of R is finite.

Theorem 16 ***Haussler and Welzl, 1986, Theorem 3.7 therein*** *Let μ be a probability measure defined over a domain $\mathcal{X}$. Let R be a collection of subsets of $\mathcal{X}$ with a finite VC dimension d. Let $\varepsilon > 0$ and assume $u \geq 8/\varepsilon$. A sample $S = \{x_i\}_{i=1}^u$ selected at random from μ^u is an ε-net for R with a probability of at least $1 - \phi(d, 2u)\, 2^{1-\varepsilon u/2}$.*

Proof of Theorem 14

By a slight abuse of notation, we use f_δ to denote both a function and a subset of $\mathcal{X}$ that includes every $x \in \mathcal{X}$ for which $D_Q(f|x) \leq D_Q^{\delta,\mu}(f)$. From Theorem 16 it follows that with probability $\geq 1 - \phi(d, 2u)\, 2^{1-\delta u/2}$ a random sample $S = \{x_i\}_{i=1}^u$ is a δ-net for $\{f_\delta\}_{f\in\mathcal{F}}$. Since for every $f \in \mathcal{F}$ we have $\mu(f_\delta) \geq \delta$ we conclude that in these cases,

$$\forall f \in \mathcal{F},\ \exists i \in [1, \ldots, u] \text{ s.t. } x_i \in f_\delta\ .$$

Note that $x_i \in f_\delta$ implies that $D_Q(f|x_i) \leq D_Q^{\delta,\mu}(f)$. Therefore, with probability $1 - \phi(d, 2u)\, 2^{1-\delta u/2}$ over the random selection of $x_1, \ldots, x_u$:

$$\forall f \in \mathcal{F},\ D_Q(f) \leq \min_i D(f|x_i) \leq D_Q^{\delta,\mu}(f)\ .$$

Let $f_1, \ldots, f_n$ be an i.i.d. sample from Q. For a fixed x_i, using Hoeffding's inequality,

$$\Pr_{f_1,\ldots,f_n}\left[\left|\frac{1}{n}\left|f_j : f_j(x_i) = 1\right| - \mu\{f : f(x_i) = 1\}\right| > \varepsilon\right] \leq 2\exp\left(-2n\varepsilon^2\right)\ .$$

Hence, with a probability of $1-u\exp\left(-2n\varepsilon^2\right)$,

$$\forall i,\ \left|\frac{1}{n}\left|f_j : f_j(x_i)=1\right| - \mu\{f\in\mathcal{F} : f(x_i)=1\}\right| \leq \varepsilon\ .$$

Clearly, in the same setting, we also have that

$$\forall i,\ \left|\frac{1}{n}\left|f_j : f_j(x_i)=-1\right| - \mu\{f\in\mathcal{F} : f(x_i)=-1\}\right| \leq \varepsilon\ .$$

Thus, with a probability of at least $1-u\exp\left(-2n\varepsilon^2\right)-\phi(d,2u)\,2^{1-\varepsilon u/2}$ over the random selection of $x_1,\ldots,x_u$ and $f_1,\ldots,f_n$ we have that

$$\forall f\in\mathcal{F},\ \hat{D}_T^S(f) \leq D_Q^{\delta,\mu}(f)+\varepsilon\ .$$

Note also, that with probability 1 there will be no i in the sample such that $D_Q(f\,|\,x_i) < D_Q^{0,\mu}(f)$. Therefore, it is also true that

$$\forall f\in\mathcal{F},\ D_Q^{0,\mu}(f)-\varepsilon \leq \hat{D}_T^S(f)$$

while it is always true that $D_Q(f) \leq D_Q^{0,\mu}(f)$. ■

In Theorem 14 we have seen that the estimated depth uniformly converges to the true depth. However, since we are interested in deep hypotheses, it suffices that the estimate is accurate for these hypotheses, as long as "shallow" hypotheses are distinguishable from the deep ones. This is the motivation for the next theorem:

Theorem 17 *Let Q be a probability measure on $\mathcal{F}$ and let μ be a probability measure on $\mathcal{X}$. Let $\varepsilon,\delta>0$. Assume $\mathcal{F}$ has a finite VC dimension $d<\infty$ and define $\phi(d,k)$ as before. Let $D=\sup_{f\in\mathcal{F}} D_Q(f)$. If S and T are chosen at random from μ^u and Q^n respectively such that $u \geq 8/\delta$ then with probability*

$$1-u\exp\left(-2n\varepsilon^2\right)-\phi(d,2u)\,2^{1-\delta u/2}$$

the following holds:

1. *For every f such that $D_Q^{\delta,\mu}(f) < D$ we have that $\hat{D}_S^T(f) \leq D_Q^{\delta,\mu}(f)+\varepsilon$*
2. *For every f we have that $\hat{D}_S^T(f) \geq D_Q^{0,\mu}(f) \geq D_Q(f)-\varepsilon$*

where $\hat{D}_T^S(f)$ is the empirical depth computed by the depth measure algorithm.

The proof is very similar to the proof of Theorem 14. The key however, is the following lemma:

Lemma 18 *Let $D=\sup_{f\in\mathcal{F}} D_Q(f)$. For every $f\in\mathcal{F}$ let f_δ be such that $f_\delta(x)=1$ if $D_Q(f\,|\,x)<D$ and $f_\delta(x)=-1$ otherwise. Let $\mathcal{F}_\delta$ be*

$$\mathcal{F}_\delta = \{f_\delta\}_{f:D_Q^{\delta,\mu}(f)<D}\ .$$

Then the VC dimension of $\mathcal{F}_\delta$ is upper bounded by the VC dimension of $\mathcal{F}$.

Proof Assume that $x_1,\ldots,x_m$ are shattered by $\mathcal{F}_\delta$. Therefore, for every sequence $y \in \{\pm 1\}^m$ there exists f^y such that f^y_δ induces the labels y on $x_1,\ldots,x_m$. We claim that for every $y \neq y'$, the function f^y and $f^{y'}$ induce different labels on $x_1,\ldots,x_m$ and hence this sample is shattered by $\mathcal{F}$. Let $y \neq y'$ and assume, w.l.o.g. that $y_i = 1$ and $y'_i = -1$. Therefore x_i is such that

$$D_Q\left(f^y \,|\, x_i\right) < D \leq D_Q\left(f^{y'} \,|\, x_i\right) \ .$$

From the definition of the depth on the point x_i, it follows that $D_Q\left(f^y \,|\, x_i\right) \neq D_Q\left(f^{y'} \,|\, x_i\right)$ if and only if $f^y(x_i) \neq f^{y'}(x_i)$. Therefore, the sample $x_1,\ldots,x_m$ being shattered by $\mathcal{F}_\delta$ implies that it is also shattered by $\mathcal{F}$. Hence, the VC dimension of $\mathcal{F}_\delta$ is bounded by the VC dimension of $\mathcal{F}$. ■

Proof of Theorem 17.

From the theory of ε-nets (see Theorem 16), and Lemma 18 it follows that with probability $1-\phi(d,2u)\,2^{1-\delta u/2}$ over the sample S, for every $f \in \mathcal{F}$ such that $D_Q^{\delta,\mu}(f) < D$ there exists x_i such that

$$D_Q\left(f \,|\, x_i\right) \leq D_Q^{\delta,\mu}(f) < D \ .$$

Therefore, with probability greater than $1-\phi(d,2u)\,2^{1-\delta u/2}$, for every f such that $D_Q^{\delta,\mu}(f) < D$ we have that $\hat{D}_S^T(f) \leq D_Q^{\delta,\mu}(f) + \varepsilon$.

To prove the second part, note that with probability 1, for every x and every f, $D_Q\left(f \,|\, x\right) \geq D_Q^{0,\mu}(f)$. Thus if $\hat{D}_S^T(f) < D_Q(f)$ it is only because the inaccuracy in the estimation $\hat{D}_T\left(f \,|\, x_i\right) = \frac{1}{n}\sum_j 1_{f_j(x_i)=f(x_i)}$. We already showed, in the proof of Theorem 14, that with probability of $1-u\exp\left(-2n\varepsilon^2\right)$ over the sample T,

$$\forall i, f, \ \left|\frac{1}{n}\sum_j 1_{f_j(x_i)=f(x_i)} - D_Q\left(f \,|\, x\right)\right| < \varepsilon \ .$$

Hence,

$$\forall f, \ \hat{D}_S^T(f) \geq D_Q^{0,\mu}(f) - \varepsilon \ .$$

■

3.1 Finding the Median

So far we discussed ways to measure the depth. We have seen that if the samples S and T are large enough then with high probability the estimated depth is accurate uniformly for all functions $f \in \mathcal{F}$.

We use these findings to present an algorithm which approximates the predicate median. Recall that the predicate median is a function f which maximizes the depth, that is $f = \arg\max_{f\in\mathcal{F}} D_Q(f)$. As an approximation, we will present an algorithm which finds a function f that maximizes the empirical depth, that is $f = \arg\max_{f\in\mathcal{F}} \hat{D}_T^S(f)$.

The intuition behind the algorithm is simple. Let $S = \{x_i\}_{i=1}^u$. A function that has large empirical depth will agree with the majority vote on these points. However, it might be the case that such a function does not exist. If we are forced to find a hypothesis that does not agree with the majority on

Algorithm 2 Median Approximation (MA)

Inputs:

- A sample $S = \{x_1, \ldots, x_u\} \in \mathcal{X}^u$ and a sample $T = \{f_1, \ldots, f_n\} \in \mathcal{F}^n$.
- a learning algorithm $\mathcal{A}$ that given a sample returns a function consistent with it if such a function exists.

Outputs:

- a function $f \in \mathcal{F}$ which approximates the predicate median, together with its depth estimation $\hat{D}_T^S(f)$

Details:

1. Foreach $i = 1, \ldots, u$ compute $p_i^+ = \frac{1}{n}\left|\left\{j : f_j(x_i) = 1\right\}\right|$ and $q_i = \min\left\{p_i^+, 1 - p_i^+\right\}$.
2. Sort $x_1, \ldots, x_u$ such that $q_1 \geq q_2 \geq \ldots \geq q_m$
3. Foreach $i = 1, \ldots, u$ let $y_i = 1$ if $p_i^+ \geq 0.5$ otherwise, let $y_i = -1$.
4. Use binary search to find i^*, the smallest i for which $\mathcal{A}$ can find a consistent function f with the sample $S^i = \{(x_k, y_k)\}_{k=i}^u$
5. If $i^* \equiv 1$ return f and depth $\hat{D} = 1 - q_1$ else return f and depth $\hat{D} = q_{i^*-1}$.

some instances, the empirical depth will be higher if these points are such that the majority vote on them wins by a small margin. Therefore, we take a sample $T = \left\{f_j\right\}_{j=1}^n$ of functions and use them to compute the majority vote on every x_i and the fraction q_i of functions which disagree with the majority vote. A viable strategy will first try to find a function that agrees with the majority votes on all the points in S. If such a function does not exist, we remove the point for which q_i is the largest and try to find a function that agrees with the majority vote on the remaining points. This process can continue until a consistent function[5] is found. This function is the maximizer of $\hat{D}_T^S(f)$. In the Median Approximation algorithm, this process is accelerated by using binary search. Assuming that the consistency algorithm requires $O(u^c)$ for some c when working on a sample of size u, then the linear search described above requires $O\left(nu + u\log(u) + u^{c+1}\right)$ operations while invoking the binary search strategy reduces the complexity to $O(nu + u\log(u) + u^c \log(u))$.

The Median Approximation (MA) algorithm is presented in Algorithm 2. One of the key advantages of the MA algorithm is that it uses a consistency oracle instead of an oracle that minimizes the empirical error. Minimizing the empirical error is hard in many cases and even hard to approximate (Ben-David et al., 2003). Instead, the MA algorithm requires only access to an oracle that is capable of finding a consistent hypothesis if one exists. For example, in the case of a linear classifier, finding a consistent hypothesis can be achieved in polynomial time by linear programming while finding a hypothesis which approximates the one with minimal empirical error is NP hard. The rest of this section is devoted to an analysis of the MA algorithm.

5. A function is defined to be consistent with a labeled sample if it labels correctly all the instances in the sample.

Theorem 19 ***The MA Theorem***

The MA algorithm (Algorithm 2) has the following properties:

1. *The algorithm will always terminate and return a function $f \in \mathcal{F}$ and an empirical depth $\hat{D}$.*
2. *If f and $\hat{D}$ are the outputs of the MA algorithm then $\hat{D} = \hat{D}_T^S(f)$.*
3. *If f is the function returned by the MA algorithm then $f = \arg\max_{f \in \mathcal{F}} \hat{D}_T^S(f)$.*
4. *Let $\varepsilon, \delta > 0$. If the sample S is taken from μ^u such that $u \geq 8/\delta$ and the sample T is taken from Q^n then with probability of at least*

$$1 - u\exp\left(-2n\varepsilon^2\right) - \phi(d, 2u)\, 2^{1-\delta u/2} \tag{5}$$

the f returned by the MA algorithm is such that

$$D_Q^{\delta,\mu}(f) \geq \sup_{g \in F} D_Q^{0,\mu}(g) - 2\varepsilon \geq \sup_{g \in F} D_Q(g) - 2\varepsilon$$

where d is the minimum between the VC dimension of $\mathcal{F}$ and the VC dimension of the class $\mathcal{F}_\delta$ defined in Theorem 14.

To prove the MA Theorem we first prove a series of lemmas. The first lemma shows that the MA algorithm will always find a function and will return it.

Lemma 20 *The MA algorithm will always return a hypothesis f and a depth $\hat{D}$*

Proof It is sufficient to show that the binary search will always find $i^* \leq u$. Therefore, it is enough to show that there exists i such that $\mathcal{A}$ will return a consistent function f with respect to S^i. To see that, recall that $S^u = \{(x_u, y_u)\}$. Therefore, the sample contains a single point x_u with the label y_u such that at least half of the functions in T are such that $f_j(x_u) = y_u$. Therefore, there exists a function f consistent with this sample. ■

The next lemma proves that the depth computed by the MA algorithm is correct.

Lemma 21 *Let f be the hypothesis that MA returned and let $\hat{D}$ be the depth returned. Then $\hat{D} = \hat{D}_T^S(f)$.*

Proof For any function g, denote by $Y(g) = \{i : g(x_i) = y_i\}$ the set of instances on which g agrees with the proposed label y_i. $\hat{D}_T^S(g)$, the estimated depth of g, is a function of $Y(g)$ given by:

$$\hat{D}_T^S(g) = \min\left(\min_{i \in Y(g)}(1 - q_i), \min_{i \notin Y(g)} q_i\right) .$$

Since the q_i's are sorted, we can further simplify this term. Letting $i_\in = \min\{i : i \in Y(g)\}$ and $i_\notin = \max\{i : i \notin Y(g)\}$, then

$$\hat{D}_T^S(g) = \min\left((1 - q_{i_\in}), q_{i_\notin}\right) .$$

In the above term, if $Y(g)$ includes all i's we consider the term $q_{i_\notin}$ to be one. Similarly, if $Y(g)$ is empty, we consider $q_{i_\in}$ to be zero.

Let f be the hypothesis returned by MA and let $\hat{D}$ be the returned computed depth. If $i^*$is the index that the binary search returned and if $i^* = 1$ then $Y(f) = [1,\ldots,u]$ and $\hat{D}_T^S(f) = 1 - q_1$ which is exactly the value returned by MA. Otherwise, if $i^* > 1$ then $i^* - 1 \notin Y(f)$ but $[i^*,\ldots,u] \subseteq Y(f)$. Since $q_{i^*-1} \leq 0.5$ but for every i' it holds that $1 - q_{i'} \geq 0.5$ so we have that $\hat{D}_T^S(f) = q_{i^*-1}$ which is exactly the value returned by FMA. ■

The next lemma shows that the MA algorithm returns the maximizer of the empirical depth.

Lemma 22 *Let f be the function that the MA algorithm returned. Then*

$$f = \arg\max_{f \in \mathcal{F}} \hat{D}_T^S(f) \ .$$

In the proof of Lemma 21 we have seen that the empirical depth of a function is a function of the set of points on which it agrees with the majority vote. We use this observation in the proof of this lemma too.

Proof Let i^* be the value returned by the binary search and let f be the function returned by the consistency oracle. If $i^* = 1$ then the empirical depth of f is the maximum possible. Hence we may assume that $i^* > 1$ and $\hat{D}_T^S(f) = q_{i^*-1}$.

For a function $g \in \mathcal{F}$, if there exists $i > i^*$ such that $g(x_i) \neq y_i$ then $\hat{D}_T^S(g) \leq q_{i-1} \leq q_{i^*-1} \leq \hat{D}_T^S(f)$. However, if $g(x_i) = y_i$ for every $i \geq i^*$ it must be that $g(x_{i^*-1}) \neq y_{i^*-1}$ or else the binary search phase in the MA algorithm would have found $i^* - 1$ or a larger set. Therefore, $\hat{D}_T^S(g) = q_{i^*-1} = \hat{D}_T^S(f)$. ■

Finally we are ready to prove Theorem 19.

Proof of the MA Theorem (Theorem 19)

Parts 1, 2 and 3 of the theorem are proven by Lemmas 20, 21 and 22 respectively. Therefore, we focus here on the last part.

Let f be the maximizer of $\hat{D}_T^S(f)$ and let $D = \sup_f D_Q^{0,\mu}(f)$. From Theorems 14 and 17 it follows that if d is at least the smaller of the VC dimension of $\mathcal{F}$ and the VC dimension of $\mathcal{F}_\delta$, then with the probability given in (5) we have that

$$\hat{D}_T^S(f) \geq \max_g \hat{D}_T^S(g) \geq \sup_g D_Q^{0,\mu}(g) - \varepsilon = D - \varepsilon \ .$$

Moreover, if $D_Q^{\delta,\mu}(f) < D$ then $\hat{D}_T^S(f) \leq D_Q^{\delta,\mu}(f) + \varepsilon$. Therefore, either

$$D_Q^{\delta,\mu}(f) \geq D$$

or

$$D_Q^{\delta,\mu}(f) \geq \hat{D}_T^S(f) - \varepsilon \geq D - 2\varepsilon$$

which completes the proof. ■

3.2 Implementation Issues

The MA algorithm is straightforward to implement provided that one has access to three oracles: (1) An oracle capable of sampling unlabeled instances $x_1,\ldots,x_u$. (2) An oracle capable of sampling hypotheses $f_1,\ldots,f_n$ from the belief distribution Q. (3) A learning algorithm $\mathcal{A}$ that returns a hypothesis consistent with the sample of instances (if such a hypothesis exists).

The first requirement is usually trivial. In a sense, the MA algorithm converts the consistency algorithm $\mathcal{A}$ to a semi-supervised learning algorithm by using this sample. The third requirement is not too restrictive. In a sense, many learning algorithms would be much simpler if they required a hypothesis which is consistent with the entire sample as opposed to a hypothesis which minimizes the number of mistakes (see, for example, Ben-David et al., 2003). The second requirement, that is sampling hypotheses, is challenging.

Sampling from continuous hypothesis classes is hard even in very restrictive cases. For example, even if Q is uniform over a convex body, sampling from it is challenging but theoretically possible (Fine et al., 2002). A closer look at the MA algorithm and the depth estimation algorithm reveals that these algorithms use the sample of functions in order to estimate the marginal $Q[Y=1|X=x]=\Pr_{g\sim Q}[g(x)=1]$. In some cases, it is possible to directly estimate this value. For example, many learning algorithms output a real value such that the sign of the output is the predicted label and the amplitude is the margin. Using a sigmoid function, this can be viewed as an estimate of $Q[Y=1|X=x]$. This can be used directly in the above algorithms. Moreover, the results of Theorem 14 and Theorem 19 apply with $\varepsilon=0$. Note that the algorithm that is used for computing the probabilities might be infeasible for run-time applications but can still be used in the process of finding the median.

Another option is to sample from a distribution Q' that approximates Q (Gilad-Bachrach et al., 2005). The way to use a sample from Q' is to reweigh the functions when computing $\hat{D}_T(f|x)$. Note that computing $\hat{D}_T(f|x)$ such that it is close to $D_Q(f|x)$ is sufficient for estimating the depth using the depth measure algorithm (Algorithm 1) and for finding the approximated median using the MA algorithm (Algorithm 2). Therefore, in this section we will focus only on computing the empirical conditional depth $\hat{D}_T(f|x)$. The following definition provides the estimate for $D_Q(f|x)$ given a sample T sampled from Q':

Definition 23 *Given a sample T and the relative density function $\frac{dQ}{dQ'}$ we define*

$$\hat{D}_{T,\frac{dQ}{dQ'}}(f)=\frac{1}{n}\sum_j \frac{dQ(f_j)}{dQ'(f_j)}1_{f_j(x)=f(x)}\ .$$

To see the intuition behind this definition, recall that $D_Q(f|x)=\Pr_{g\sim Q}[g(x)=f(x)]$ and $\hat{D}_T(f|x)=\frac{1}{n}\sum_j 1_{f_j(x)=f(x)}$ where $T=\{f_j\}_{j=1}^n$. If T is sampled from Q^n we have that

$$E_{T\sim Q^n}\left[\hat{D}_T(f|x)\right] \;=\; \frac{1}{n}\sum_j E\left[1_{f_j(x)=f(x)}\right]=\frac{1}{n}\sum_j \Pr[f_j(x)=f(x)]=D_Q(f|x)\ .$$

Therefore, we will show that $\hat{D}_{T,\frac{dQ}{dQ'}}(f)$ is an unbiased estimate of $D_Q(f|x)$ and that it is concentrated around its expected value.

Theorem 24 *Let Q and Q' be probability measures over $\mathcal{F}$. Then:*

1. *For every* f, $E_{T\sim Q^n}\left[\hat{D}_{T,\frac{dQ}{dQ'}}(f)\right] = D_Q(f\,|\,x)$

2. *If* $\frac{dQ}{dQ'}$ *is bounded such that* $\frac{dQ}{dQ'} \leq c$ *then*

$$\Pr_{T\sim Q^n}\left[\left|\hat{D}_{T,\frac{dQ}{dQ'}}(f) - D_Q(f\,|\,x)\right| > \varepsilon\right] < 2\exp\left(-\frac{2n\varepsilon^2}{c^2}\right) .$$

Proof To prove the first part we note that

$$\begin{aligned} E_{T\sim Q'^n}\left[\hat{D}_{T,\frac{dQ}{dQ'}}(f)\right] &= E_{T\sim Q'^n}\left[\frac{1}{n}\sum_j \frac{dQ(f_j)}{dQ'(f_j)} 1_{f_j(x)=f(x)}\right] \\ &= E_{g\sim Q'}\left[\frac{dQ(g)}{dQ'(g)} 1_{g(x)=f(x)}\right] = \int_g \frac{dQ(g)}{dQ'(g)} 1_{g(x)=f(x)} dQ'(g) \\ &= \int_g 1_{g(x)=f(x)} dQ(g) = D_Q(f\,|\,x) \ . \end{aligned}$$

The second part is proved by combining Hoeffding's bound with the first part of this theorem. ■

3.3 Tukey Depth and Median Algorithms

To complete the picture we demonstrate how the algorithms presented here apply to the problems of computing Tukey's depth function and finding the Tukey median. In section 2.1 we showed how to cast the Tukey depth as a special case of the predicate depth. We can use this reduction to use Algorithm 1 and Algorithm 2 to compute the Tukey depth and approximate the median respectively. To compute these values, we assume that one has access to a sample of points in $\mathbb{R}^d$, which we denote by $f_1,\ldots,f_n$. We also assume that one has access to a sample of directions and biases of interest. That is, we assume that one has access to a sample of x_i's such that $x_i \in \mathbb{S}\times\mathbb{R}$ where $\mathbb{S}$ is the unit sphere. Hence, we interpret x_i as a combination of a d-dimensional unit vector x_i^v and an offset term x_i^θ. Algorithm 3 shows how to use these samples to estimate the Tukey depth of a point $f\in\mathbb{R}^d$. Algorithm 4 shows how to use these samples to approximate the Tukey median. The analysis of these algorithms follows from Theorems 17 and 19 recalling that the VC dimension of this problem is d.

Computing the Tukey depth requires finding the infimum over all possible directions. As other approximation algorithm do (see Section 4.5) the algorithm presented here finds a minimum over a sample of possible directions represented by the sample S. When generating this sample, it is natural to select x_i^v uniformly from the unit sphere. According to the algorithms presented here one should also select x_i^θ at random. However, for the special case of the linear functions we study here, it is possible to find the minimal depth over all possible selections of x_i^θ once x_i^v is fixed. This can be done by counting the number of f_j's such that $f_j\cdot x_i^v > f\cdot x_i^v$ and the number of f_j's such that $f_j\cdot x_i^v < f\cdot x_i^v$ and taking the minimal value between these two. We use this in the algorithm presented here.

Algorithm 4 selects a set of random directions $x_1,\ldots,x_u$. The median f should be central in every direction. That is, if we project $f_1,\ldots,f_n$ and f on x_i then the projection of f should be close

Algorithm 3 Tukey Depth Estimation

Inputs:

- A sample $S = \{x_1, \ldots, x_u\}$ such that $x_i \in \mathbb{S}$
- A sample $T = \{f_1, \ldots, f_n\}$ such that $f_j \in \mathbb{R}^d$
- A point $f \in \mathbb{R}^d$

Output:

- $\hat{D}_T^S(f)$ - an approximation for the depth of f

Algorithm:

1. for $i = 1, \ldots, u$ compute $\hat{D}_T(f \,|\, x_i) = \frac{1}{n} \min\left(\left|f_j : f_j \cdot x_i > f \cdot x_i\right|, \left|f_j : f_j \cdot x_i < f \cdot x_i\right|\right)$
2. return $\hat{D}_T^S(f) = \min_i \hat{D}(f \,|\, x_i)$

Algorithm 4 Tukey Median Approximation

Inputs:

- A sample $S = \{x_1, \ldots, x_u\}$ such that $x_i \in \mathbb{S}$
- A sample $T = \{f_1, \ldots, f_n\}$ such that $f_j \in \mathbb{R}^d$
- A linear programs solver $\mathcal{A}$ that given a set of linear constraints finds a point that is consistent with the constraints if such a point exists.

Outputs:

- A point $f \in \mathbb{R}^d$ and its depth estimation $\hat{D}_T^S(f)$

Details:

1. Foreach $i = 1, \ldots, u$ and $j = 1, \ldots, n$ compute $f_j \cdot x_i$
2. Let $s_i^1, \ldots s_i^n$ be the sorted values of $f_j \cdot x_i$.
3. Use binary search to find the smallest $k = 0, \ldots, n/2$ for which $\mathcal{A}$ can find f such that

$$\forall i \;\; s_i^{\lfloor \frac{n}{2} \rfloor - k} \leq f \cdot x_i < s_i^{\lceil \frac{n}{2} \rceil + k}$$

4. Return the f that $\mathcal{A}$ found for the smallest k in (3).

to the median of the projection, that is, it should have high one dimensional depth. Therefore, we can start by seeking f with the highest possible depth in every direction. If such f does not exist we can weaken the depth requirement in each direction and try again until we can find a candidate f. Algorithm 4 accelerates this process by using binary search. Note that since the above procedures use only inner products, kernel versions are easily constructed.

4. Relation to Previous Work

In this section we survey the relevant literature. Since depth plays an important role in multivariate statistics it has been widely studied, see Liu et al. (1999), for example, for a comprehensive introduction to statistical depth and its applications in statistics and the visualization of data. We focus only on the part that is related to the work presented here. To make this section easier to follow, we present each related work together with its contexts. Note however that the rest of this work does not build upon information presented in this section and thus a reader can skip this section if he wishes to do so.

4.1 Depth for Functional Data

López-Pintado and Romo (2009) studied depth for functions. The definitions of depth used therein is closer in spirit to the simplicial depth in the multivariate case (Liu, 1990). As a consequence it is defined only for the case where the measure over the function space is an empirical measure over a finite set of functions. Zuo (2003) studied the projection based depth. For a point x in $\mathbb{R}^d$ and a measure ν over $\mathbb{R}^d \times \mathbb{R}$, the depth of x with respect to ν is defined to be

$$\begin{aligned} D_\nu(x) &= \left(1 + \sup_{\|u\|=1} \phi(u,x,\nu)\right)^{-1} \quad \text{where} \\ \phi(u,x,\nu) &\equiv \frac{|u \cdot x - \mu(\nu|_x)|}{\sigma(\nu|_x)} \end{aligned}$$

where μ is a measure of dislocation and σ is a measure of scaling. The functional depth we present in this work can be presented in similar form by defining, for a function f,

$$\begin{aligned} D(f,\nu) &= \left(1 + \sup_{x \in X} \phi(f,x,\nu)\right)^{-1} \quad \text{where} \\ \phi(f,x,\nu) &= |f(x) - E_{g\sim\nu}[g(x)]| \ . \end{aligned}$$

Fraiman and Muniz (2001) introduced an extension of univariate depth to function spaces. For a real function f, the depth of f is defined to be $E_x[D(f(x))]$ where $D(\cdot)$ is the univariate depth function. It is not clear how to use this definition in the binary classification setting. Since the range of the functions contains only two possible values, the univariate rank is of limited utility. However, if we choose the rank function such that the rank of a value is the probability that a function will assign this value, we arrive at a similar definition to the one we propose. The main difference is that Fraiman and Muniz (2001) define the depth as an average over all x's, while in our setting we take the infimum. This plays a key role in our analysis.

4.2 Depth and Classification

Ghosh and Chaudhuri (2005a) used depth for classification purposes. Given samples of data from the different classes, one creates depth functions for each of the classes. At inference time, the depth of a point x is computed with respect to each of the samples. The algorithm associates an instance x with the class in which x is deepest. Ghosh and Chaudhuri (2005a) prove generalization bounds in the case in which each class has a elliptic distribution. Cuevas et al. (2007) used a similar approach and compared the performance of different depth functions empirically. Jörnsten (2004) used a similar approach with an L_1 based depth function. Billor et al. (2008) proposed another variant of this technique.

Ghosh and Chaudhuri (2005b) introduced two variants of depth functions to be used for learning linear classifiers. Let $\{(x_i, y_i)\}_{i=1}^n$ be the training data such that $x_i \in \mathbb{R}^d$ and $y_i \in \pm 1$. In the first variant, the depth of a linear classifier $\alpha \in \mathbb{R}^d$ is defined to be

$$U(\alpha) = \frac{1}{n^+ n^-} \sum_{i:y_i=1} \sum_{j:y_j=-1} \mathbb{I}[\alpha \cdot (x_i - x_j) > 0]$$

where n^+ and n^- are the numbers of positive (and negative) examples, and $\mathbb{I}$ is the indicator function. The regression based depth function is defined to be

$$\Delta(\alpha, \beta) = \frac{\pi^+}{n^+} \sum_{i:y_i=1} \mathbb{I}[\alpha \cdot x_i + \beta > 0] + \frac{\pi^-}{n^-} \sum_{i:y_i=-1} \mathbb{I}[\alpha \cdot x_i + \beta < 0]$$

where π^+ and π^- are positive scalars that sum to one. It is easy to see that the regression based depth defined here is the balanced misclassification probability. The authors showed that as the sample size goes to infinity, the maximal depth classifier is the optimal linear classifier. However, since minimizing this quantity is known to be hard, the authors suggesting using the logistic function as a surrogate to the indicator function. Therefore, these methods are very close (and in some cases identical) to logistic regression.

Gilad-Bachrach et al. (2004) used the Tukey depth to analyze the generalization performance of the Bayes Point Machine (Herbrich et al., 2001). This work uses depth in a similar fashion to the way we use it in the current study. However, the definition of the Bayes depth therein compares the generalization error of a hypothesis to the Bayes classifier in a way that does not allow the use of the PAC-Bayes theory to bound the gap between the empirical error and the generalization error. As a result the analysis in Gilad-Bachrach et al. (2004) was restricted to the realizable case in which the empirical error is zero.

4.3 Regression Depth

Rousseeuw and Hubert (1999) introduced the notion of regression depth. They discussed linear regression but their definition can be extended to general function classes in the following way: Let $\mathcal{F} = \{f : \mathcal{X} \mapsto \mathbb{R}\}$ be a function class and let $S = \{(x_i, y_i)\}_{i=1}^n$ be a sample such that $x_i \in \mathcal{X}$ and $y_i \in \mathbb{R}$. We say that the function $f \in \mathcal{F}$ has depth zero ("non-fit" in Rousseeuw and Hubert, 1999) if there exists $g \in \mathcal{F}$ that is strictly better than f on every point in S. That is, for every point (x_i, y_i) one of the following applies:

$$\begin{aligned} &i. \quad f(x_i) < g(x_i) \leq y_i \\ &ii. \quad f(x_i) > g(x_i) \geq y_i \ . \end{aligned}$$

A function $f \in \mathcal{F}$ is said to have a depth d if d is the minimal number of points that should be removed from S to make f a non-fit.

Christmann (2006) applied the regression depth to the classification task. He used the logit function to convert the classification task to a regression problem. He showed that in this setting the regression depth is closely related to the logistic regression problem and the well known risk minimization technique.

4.4 An Axiomatic Definition of Depth

Most of the applications of depth for classification define the depth of a classifier with respect to a given sample. This is true for the regression depth as well. In that respect, the empirical accuracy of a function is a viable definition of depth. However, in this study we define the depth of a function with respect to a probability measure over the function class. Following Zuo and Serfling (2000) we introduce a definition of a depth function for the classification setting. Our definition has four conditions, or axioms. The first condition is an invariance requirement, similar to the affine invariance requirement in multivariate depth functions. In our setting, we require that if there is a symmetry group acting on the hypothesis class then the same symmetry group acts on the depth function too. The second condition is a symmetry condition: it requires that if the Bayesian optimal hypothesis happens to be a member of the hypothesis class then the Bayesian optimal hypothesis is the median hypothesis. The third condition is the monotonicity condition. This requires that if f_1 and f_2 are two hypotheses such that f_1 is strictly closer to the Bayesian optimal hypothesis than f_2, then f_1 is deeper than f_2. The final requirement is that the depth function is not trivial, that is, that the depth is not a constant value.

Definition 25 *Let $\mathcal{F} = \{f : \mathcal{X} \mapsto \pm 1\}$ and let Q be a probability measure over $\mathcal{F}$. $D_Q : \mathcal{F} \mapsto \mathbb{R}^+$ is a depth function if it has the following properties:*

1. ***Invariance:*** *If $\sigma : \mathcal{X} \mapsto \mathcal{X}$ is a symmetry of $\mathcal{F}$ in the sense that for every $f \in \mathcal{F}$ there exists $f_\sigma \in \mathcal{F}$ such that $f(\sigma(x)) = f_\sigma(x)$ then for every Q and f: $D_Q(f) = D_{Q_\sigma}(f_\sigma)$. Here, Q_σ is such that for every measurable set $A \subseteq \mathcal{F}$ we let $A_\sigma = \{f_\sigma : f \in A\}$ and have that $Q(A) = Q_\sigma(A_\sigma)$.*
2. ***Symmetry:*** *if there exists $f^* \in \mathcal{F}$ such that $\forall x \in \mathcal{X},\ Q\{f : f(x) = f^*(x)\} \geq 1/2$ then $D_Q(f^*) = \sup_f D_Q(f)$.*
3. ***Monotonicity:*** *if there exists $f^* \in \mathcal{F}$ such that $D_Q(f^*) = \sup_f D_Q(f)$ then for every $f_1, f_2 \in \mathcal{F}$, if $f_1(x) \neq f^*(x) \Longrightarrow f_2(x) \neq f^*(x)$ then $D_Q(f_1) \geq D_Q(f_2)$.*
4. ***Non-trivial:*** *for every $f \in \mathcal{F}$, there exist Q such that f is the unique maximizer of D_Q.*

Our definition attempts to capture the same properties that Zuo and Serfling (2000) considered, with a suitable adjustment for the classification setting. It is a simple exercise to verify that the predicate depth meets all of the above requirements.

4.5 Methods for Computing the Tukey Median

Part of the contribution of this work is the proposal of algorithms for approximating the predicate depth and the predicate median. The Tukey depth is a special case of the predicate depth and therefore we survey the existing literature for computing the Tukey median here. Chan (2004) presented

optimal algorithms for computing the Tukey median. Chan presented a randomized algorithm that can find the Tukey median for a sample of n points with expected computational complexity of $O(n \log n)$ when the data is in $\mathbb{R}^2$ and $O\left(n^{d-1}\right)$ when the data is in $\mathbb{R}^d$ for $d > 2$. It is conjectured that these results are optimal for finding the exact median. Massé (2002) analyzed the asymptotic behavior of the empirical Tukey depth. The empirical Tukey depth is the Tukey depth function when it is applied to an empirical measure sampled from the true distribution of interest. He showed that the empirical depth converges uniformly to the true depth with probability one. Moreover, he showed that the empirical median converges to the true median at a rate that scales as $1/\sqrt{n}$. Cuesta-Albertos and Nieto-Reyes (2008) studied the random Tukey depth. They proposed picking k random directions and computing the univariate depth of each candidate point for each of the k directions. They defined the random Tukey depth for a given point to be the minimum univariate depth of this point with respect to the k random directions. In their study, they empirically searched for the number of directions needed to obtain a good approximation of the depth. They also pointed out that the random Tukey depth uses only inner products and hence can be computed in any Hilbert space.

Note that the empirical depth of Massé (2002) and the random Tukey depth of Cuesta-Albertos and Nieto-Reyes (2008) are different quantities. In the empirical depth, when evaluating the depth of a point x, one considers every possible hyperplane and evaluates the measure of the corresponding half-space using only a sample. On the other hand, in the case of random depth, one evaluates only k different hyperplanes. However, for each hyperplane it is assumed that the true probability of the half-space is computable. Therefore, each one of these approaches solves one of the problems involved in computing the Tukey depth. However, in reality, both problems need to be solved simultaneously. That is, since scanning all possible hyperplanes is computationally prohibited, one has to find a subset of representative hyperplanes to consider. At the same time, for each hyperplane, computing the measure of the corresponding half-space is prohibitive for general measures. Thus, an approximation is needed here as well. The solution we present addresses both issues and proves the convergence of the outcome to the Tukey depth, as well as giving the rate of convergence.

Since finding the deepest point is hard, some studies focus on just finding a deep point. Clarkson et al. (1996) presented an algorithm for finding a point with depth $\Omega(1/d^2)$ in polynomial time. For the cases in which we are interested, this could be insufficient. When the distribution is log-concave, there exists a point with depth $1/e$, independent of the dimension (Caplin and Nalebuff, 1991). Moreover, for any distribution there is a point with a depth of at least $1/d+1$ (Carathèodory's theorem).

4.6 PAC-Bayesian Bounds

Our works builds upon the PAC-Bayesian theory that was first introduced by McAllester (1999). These results were further improved in a series of studies (see, for example, Seeger, 2003; Ambroladze et al., 2007; Germain et al., 2009). These results bound, with high probability, the gap between the empirical error of a stochastic classifier based on a posterior Q to the expect error of this classifier in terms of the KL-divergence between Q and the prior P. Some of these studies demonstrate how this technique can be applied to the class of linear classifier and how to improve the bounds by using parts of the training data to learn a prior P to further tighten the generalization bounds.

In the current study we use different approaches which result in different type of bounds. Theorem 4 shows a multiplicative bound on the error of a classifier with respect to the error of the

Gibbs classifier. For example, if the posterior is log-concave and the hypothesis is the mean of the posterior, then the multiplicative factor is $e \cong 2.71$. This bound contains no additive components, therefore, if the generalization error of the Gibbs classifier is small, this new bound may be superior compared to bounds which have additive components (Ambroladze et al., 2007; Germain et al., 2009). The structure of this bound is closer to the consistency bounds for the Nearest Neighbor algorithm (Fix and Hodges Jr, 1951). However, unlike consistency bounds, the bound of Theorem 4 applies to any sample size and any method of obtaining the data.

Another aspect of Theorem 4 is that the bound applies to any classification function. That means that it does not assume that the classifier comes from the same class on which the Gibbs classifier is defined, neither does it make any assumptions on the training process. For example, the training error does not appear in this bound.

Theorem 5 uses the PAC-Bayesian theory to relate the training error of the Gibbs classifier to the generalization error of a deep classifier. In Ambroladze et al. (2007) and Germain et al. (2009) the posterior Q is chosen to be a unit variance Gaussian around the linear classifier of interest. Using the same posterior in Theorem 5 will result in inferior results since there is an extra penalty of factor 2 due to the $1/2$ depth of the center of the Gaussian. However, our bound provides more flexibility in choosing the posterior Q in the tradeoff between the empirical error, the KL divergence and the depth. It is left as an open problem to determine if one can derive better bounds by using this flexibility.

5. Discussion

In this study we addressed the hypothesis selection problem. That is, given a posterior belief over the hypothesis class, we examined the problem of choosing the best hypothesis. To address this challenge, we defined a depth function for classifiers, the predicate depth, and showed that the generalization of a classifier is tied to its predicate depth. Therefore, we suggested that the deepest classifier, the predicate median, is a good candidate hypothesis to select. We analyzed the breakdown properties of the median and showed it is related to the depth as well. We contrasted these results with the more commonly used maximum a posteriori classifier.

In the second part of this work we discussed the algorithmic aspects of our proposed solution. We presented efficient algorithms for uniformly measuring the predicate depth and for finding the predicate median. Since the Tukey depth is a special case of the depth presented here, it also follows that the Tukey depth and the Tukey median can be approximated in polynomial time by our algorithms.

Our discussion was limited to the binary classification case. It will be interesting to see if this work can be extended to other scenarios, for example, regression, multi-class classification and ranking. These are open problems at this point.

References

A. Ambroladze, E. Parrado-Hernández, and J. Shawe-Taylor. Tighter PAC-Bayes bounds. *Advances in neural information processing systems*, 19:9, 2007.

S. Ben-David, N. Eiron, and P. M. Long. On the difficulty of approximately maximizing agreements. *J. Comput. Syst. Sci.*, 66(3):496–514, 2003.

N. Billor, A. Abebe, A. Turkmen, and S. V. Nudurupati. Classification based on depth transvariations. *Journal of classification*, 25(2):249–260, 2008.

C. Borell. Convex set functions in d-space. *Periodica Mathematica Hungarica*, 6:111–136, 1975. ISSN 0031-5303. 10.1007/BF02018814.

A. Caplin and B. Nalebuff. Aggregation and social choice: A mean voter theorem. *Econometrica*, 59(1):1–23, January 1991.

T. M. Chan. An optimal randomized algorithm for maximum Tukey depth. In *SODA*, pages 430–436, 2004.

A. Christmann. Regression depth and support vector machine. *DIMACS series in discrete mathematics and theoretical computer science*, 72:71–86, 2006.

K. L. Clarkson, D. Eppstein, G. L. Miller, C. Sturtivant, and S. H. Teng. Approximating center points with iterative radon points. *Int. J. Comput. Geometry Appl.*, 6(3):357–377, 1996.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

J. A. Cuesta-Albertos and A. Nieto-Reyes. The random Tukey depth. *Computational Statistics & Data Analysis*, 52, 2008.

A. Cuevas, M. Febrero, and R. Fraiman. Robust estimation and classification for functional data via projection-based depth notions. *Computational Statistics*, 2007.

P. L. Davies and U. Gather. Breakdown and groups. *The Annals of Statistics*, 33(3):977–1035, 2005.

S. Fine, R. Gilad-Bachrach, and E. Shamir. Query by committee, linear separation and random walks. *Theor. Comput. Sci.*, 284(1):25–51, 2002.

E. Fix and J. L. Hodges Jr. Discriminatory analysis-nonparametric discrimination: Consistency properties, 1951.

R. Fraiman and G. Muniz. Trimmed means for functional data. *Test*, 2001.

P. Germain, A. Lacasse, F. Laviolette, M. Marchand, and S. Shanian. From PAC-Bayes bounds to KL regularization. In *ICML*, 2009.

A. K. Ghosh and P. Chaudhuri. On maximum depth and related classifiers. *Scandinavian Journal of Statistics*, 32(2):327–350, 2005a.

A. K. Ghosh and P. Chaudhuri. On data depth and distribution-free discriminant analysis using separating surfaces. *Bernoulli*, 11(1):1–27, 2005b.

R. Gilad-Bachrach, A. Navot, and N. Tishby. Bayes and Tukey meet at the center point. In *Proceedings of the Conference on Learning Theory (COLT)*, pages 549–563. Springer, 2004.

R. Gilad-Bachrach, A. Navot, and N. Tishby. Query by committee made real. In *NIPS*, 2005.

F. R. Hampel. A general qualitative definition of robustness. *The Annals of Mathematical Statistics*, 42(6):1887–1896, 1971.

D. Haussler and E. Welzl. Epsilon-nets and simplex range queries. In *Symposium on Computational Geometry*, pages 61–71, 1986. doi: 10.1145/10515.10522.

X. He. Discussion: Breakdown and groups. *The Annals of Statistics*, 33(3):998–1000, 2005.

R. Herbrich, T. Graepel, and C. Campbell. Bayes point machines. *The Journal of Machine Learning Research*, 1:245–279, 2001.

R. Jörnsten. Clustering and classification based on the L1 data depth. *Journal of Multivariate Analysis*, 90, 2004.

R. Y. Liu. On a notion of data depth based on random simplices. *Annals of Statistics*, 1990.

R. Y. Liu, J. M. Parelius, and K. Singh. Multivariate analysis by data depth: descriptive statistics, graphics and inference,(with discussion and a rejoinder by Liu and Singh). *The Annals of Statistics*, 27(3):783–858, 1999.

S. López-Pintado and J. Romo. On the concept of depth for functional data. *Journal of The American Statistical Association*, 104, 2009.

J. C. Massé. Asymptotics for the Tukey median. *Journal of Multivariate Analysis*, 81, 2002.

D. A. McAllester. Some PAC-Bayesian theorems. *Machine Learning*, 37(3):355–363, 1999.

P. J. Rousseeuw and M. Hubert. Regression depth. *Journal of the American Statistical Association*, 94(446):388–402, 1999.

M. Seeger. PAC-Bayesian generalisation error bounds for gaussian process classification. *The Journal of Machine Learning Research*, 3:233–269, 2003.

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

J. Tukey. Mathematics and the picturing of data. In *Proceedings of the International Congress of Mathematicians*, 1975.

Y. Zuo. Projection-based depth functions and associated medians. *The Annals of Statistics*, 2003.

Y. Zuo and R. Serfling. General notions of statistical depth function. *The Annals of Statistics*, 28 (2):461–482, 2000.

Journal of Machine Learning Research 14 (2013) 3619-3647 Submitted 4/13, Revised 10/13; Published 12/13

A Max-Norm Constrained Minimization Approach to 1-Bit Matrix Completion

Tony Cai TCAI@WHARTON.UPENN.EDU
Statistics Department
The Wharton School
University of Pennsylvania
Philadelphia, PA 19104, USA

Wen-Xin Zhou WENXIN.ZHOU@UNIMELB.EDU.AU
Department of Mathematics and Statistics
University of Melbourne
Parkville, VIC 3010, Australia

Editor: Nathan Srebro

Abstract

We consider in this paper the problem of noisy 1-bit matrix completion under a general non-uniform sampling distribution using the max-norm as a convex relaxation for the rank. A max-norm constrained maximum likelihood estimate is introduced and studied. The rate of convergence for the estimate is obtained. Information-theoretical methods are used to establish a minimax lower bound under the general sampling model. The minimax upper and lower bounds together yield the optimal rate of convergence for the Frobenius norm loss. Computational algorithms and numerical performance are also discussed.

Keywords: 1-bit matrix completion, low-rank matrix, max-norm, trace-norm, constrained optimization, maximum likelihood estimate, optimal rate of convergence

1. Introduction

Matrix completion, which aims to recover a low-rank matrix from a subset of its entries, has been an active area of research in the last few years. It has a range of successful applications. In some real-life situations, however, the observations are highly *quantized*, sometimes even to a single bit and thus the standard matrix completion techniques do not apply. Take the Netflix problem as an example, the observations are the ratings of movies, which are quantized to the set of integers from 1 to 5. In the more extreme case such as recommender systems, only a single bit of rating standing for a "thumbs up" or "thumbs down" is recorded at each occurrence. Another example of applications is targeted advertising, such as the relevance of advertisements on Hulu. Each user who is watching TV shows on Hulu is required to answer yes/no to the question*"Is this ad relevant to you?"*. Noise effect should be considered since there are users who just click no to all the advertisements. In general, people would prefer to have advertisement catered to them, rather than to endure random advertisement. Targeted marketing that uses customer needs tends to serve better than random, scattershot advertisements. Similar idea has already been employed in mail system (Goldberg et al., 1992). Other examples from recommender systems include rating music on Pandora and posts on Reddit or MathOverflow, in which each observation consists of a single bit representing a positive

or negative rating. Similar problem also arises in analyzing incomplete survey designs containing simple agree/disagree questions in the analysis of survey data, and distance matrix recovery in multidimensional scaling that incorperates binary responses with incomplete data (Green and Wind, 1973; Spence and Demoney, 1974). See Davenport et al. (2012) for more discussions on potential applications.

Motivated by these applications, Davenport et al. (2012) considered the *1-bit matrix completion problem* of recovering an approximately low-rank matrix $M^* \in \mathbb{R}^{d_1 \times d_2}$ from a set of n noise corrupted sign (1-bit) measurements. In particular, they proposed a trace-norm constrained maximum likelihood estimator to estimate M^*, based on a small number of binary samples observed according to a probability distribution determined by the entries of M^*. It was also shown that the trace-norm constrained optimization method is minimax rate-optimal under the uniform sampling model. This problem is closely connected to and in some respects more challenging than the *1-bit compressed sensing*, which was introduced and first studied in Boufounos and Baraniuk (2008). The 1-bit measurements are meant to model quantization in the extreme case, and a surprising fact is that when the signal-to-noise ratio is low, empirical evidence demonstrates that such extreme quantization can be optimal when constrained to a fixed bit budget (Laska and Baraniuk, 2012). See Plan and Vershynin (2013a) for the recent results and references on 1-bit compressed sensing.

To be more specific, consider an arbitrary unknown $d_1 \times d_2$ target matrix M^* with rank at most r. Suppose a subset $S = \{(i_1, j_1), ..., (i_n, j_n)\}$ of entries of a binary matrix Y is observed, where the entries of Y depend on M^* in the following way:

$$Y_{i,j} = \begin{cases} +1, & \text{if } M^*_{i,j} + Z_{i,j} \geq 0, \\ -1, & \text{if } M^*_{i,j} + Z_{i,j} < 0. \end{cases}$$

Here $Z = (Z_{ij}) \in \mathbb{R}^{d_1 \times d_2}$ is a general noise matrix. This latent variable matrix model can been seen as a direct analogue to the usual 1-bit compressed sensing model, in which only the signs of measurements are observed. It is known that an s-sparse signal can still be approximately recovered from $O(s\log(d/s))$ random linear measurements. See, for example, Jacques et al. (2011), Plan and Vershynin (2013a), Plan and Vershynin (2013b) and Ai et al. (2013).

Contrary to the standard matrix completion model and many other statistical problems, random noise turns out to be helpful and has a positive effect in the 1-bit case, since the problem is ill-posed in the absence of noise as described in Davenport et al. (2012). In particular, when $Z = 0$ and $M^* = uv^T$ for some vectors $u \in \mathbb{R}^{d_1}, v \in \mathbb{R}^{d_2}$ having no zero coordinates, then the radically disparate matrix $\widetilde{M} = \text{sign}(u)\text{sign}^T(v)$ will lead to the same observations Y. Thus M and $\widetilde{M}$ are indistinguishable. However, it has been surprisingly noticed that the problem may become well-posed when there are some additional stochastic variations, that is, $Z \neq 0$ is an appropriate random noise matrix. This phenomenon can be regarded as a "dithering" effect brought by random noise.

Although the trace-norm constrained optimization method has been shown to be minimax rate-optimal under the uniform sampling model, it remains unclear that the trace-norm is the best convex surrogate to the rank. A different convex relaxation for the rank, the matrix *max-norm*, has been duly noted in machine learning literature since Srebro et al. (2005), and it was shown to be empirically superior to the trace-norm for collaborative filtering problems. Regarding a real $d_1 \times d_2$ matrix as an operator that maps from $\mathbb{R}^{d_2}$ to $\mathbb{R}^{d_1}$, its rank can be alternatively expressed as the smallest integer k, such that it is possible to express $M = UV^T$, where $U \in \mathbb{R}^{d_1 \times k}$ and $V \in \mathbb{R}^{d_2 \times k}$. In terms of the matrix factorization $M = UV^T$, we would like U and V to have a small number of columns. The number of columns of U and V can be relaxed in a different way from the usual trace-norm by the

so-called *max-norm* (Linial et al., 2007), which is defined by

$$\|M\|_{\max} = \min_{M=UV^T} \left\{ \|U\|_{2,\infty} \|V\|_{2,\infty} \right\}, \tag{1}$$

where the infimum is over all factorizations $M = UV^T$ with $\|U\|_{2,\infty}$ being the operator norm of $U : \ell_2^k \to \ell_\infty^{d_1}$ and $\|V\|_{2,\infty}$ the operator norm of $V : \ell_2^k \to \ell_\infty^{d_2}$ (or, equivalently, $V^T : \ell_1^{d_2} \to \ell_2^k$) and $k = 1, ..., \min(d_1, d_2)$. It is not hard to check that $\|U\|_{2,\infty}$ is equal to the largest ℓ_2 norm of the rows in U. Since ℓ_2 is a Hilbert space, $\|\cdot\|_{\max}$ indeed defines a norm on the space of operators between $\ell_1^{d_2}$ and $\ell_\infty^{d_1}$. Comparably, the trace-norm has a formulation similar to (1), as given below in Section 2.1.

Foygel and Srebro (2011) first used the max-norm for matrix completion under the uniform sampling distribution. Their results are direct consequences of a recent bound on the excess risk for a smooth loss function, such as the quadratic loss, with a bounded second derivative (Srebro et al., 2010). Matrix completion under a non-degenerate random sampling model was studied in Cai and Zhou (2013), where it was shown that the max-norm constrained minimization method is rate-optimal and it yields a more stable approximate recovery guarantee, with respect to the sampling distributions, than trace-norm based approaches.

Davenport et al. (2012) analyzed 1-bit matrix completion under the *uniform sampling model*, where observed entries are assumed to be sampled randomly and uniformly. In such a setting, the trace-norm constrained approach has been shown to achieve minimax rate of convergence. However, in certain application such as collaborative filtering, the uniform sampling model is over idealized. In the Netflix problem, for instance, the uniform sampling model is equivalent to assuming all users are equally likely to rate every movie and all movies are equally likely to be rated by any user. In practice, inevitably some users are more active than others and some movies are more popular and thus rated more frequently. Therefore, the sampling distribution is in fact non-uniform. In this scenario, Salakhutdinov and Srebro (2010) showed that the standard trace-norm relaxation can behave very poorly, and suggested to use a weighted variant of the trace-norm, which takes the sampling distribution into account. Since the true sampling distribution is most likely unknown and can only be estimated based on the locations of those entries that are revealed in the sample, what commonly used in practice is the empirically-weighted trace norm. Foygel et al. (2011) provided rigorous recovery guarantees for learning with the standard weighted, smoothed weighted and smoothed empirically-weighted trace-norms. In particular, they gave upper bounds on excess error, which show that there is no theoretical disadvantage of learning with smoothed empirical marginals as compared to learning with smoothed true marginals.

In this paper, we study matrix completion based on noisy 1-bit observations under a general (non-degenerate) sampling model using the max-norm as a convex relaxation for the rank. The rate of convergence for the max-norm constrained maximum likelihood estimate is obtained. A matching minimax lower bound is established under the general non-uniform sampling model using information-theoretical methods. The minimax upper and lower bounds together yield the optimal rate of convergence for the Frobenius norm loss. As a comparison with the max-norm constrained optimization approach, we also analyze the recovery guarantee of the weighted trace-norm constrained method in the setting of non-uniform sampling distributions. Our result includes an additional logarithmic factor, which might be an artifact of the proof technique. The numerical results in Section 5 show that, even when the sampling distribution is uniform, the max-norm based regularization might slightly outperform the corresponding trace-norm method. To sum up, the max-norm regularized approach indeed provides a unified and stable approximate recovery guarantee with re-

spect to the sampling distributions, while previously used approaches are based on different variants of the trace-norm which may sometimes seem artificial to practitioners.

When the noise distribution is Gaussian or more generally log-concave, the negative log-likelihood function for M, given the measurements, is convex, hence computing the max-norm constrained maximum likelihood estimate is a convex optimization problem. The computational effectiveness of this method is also studied, based on a first-order algorithm developed in Lee et al. (2010) for solving convex programs involving a max-norm constraint, which outperforms the semi-definite programming method used in Srebro et al. (2005). It will be shown in Section 4 that the convex optimization problem can be implemented in polynomial time as a function of the sample size and the matrix dimensions.

The rest of the paper is organized as follows. Section 2 begins with the basic notation and definitions, and then states a collection of useful results on the matrix norms, Rademacher complexity and distances between matrices that will be needed throughout the paper. Section 3 introduces the 1-bit matrix completion model and the estimation procedure and investigates the theoretical properties of the estimator. Both minimax upper and lower bounds are established. The results show that the max-norm constraint maximum likelihood estimator is rate-optimal over the parameter space. Section 3 also gives a comparison of our results with previous work. Computational algorithms are discussed in Section 4, and numerical performance of the proposed algorithm is presented in Section 5. The paper is concluded with a brief discussion in Section 6, and the proofs of the main results are given in Section 7.

2. Notations and Preliminaries

In this section, we introduce basic notation and definitions that will be used throughout the paper, and state some known results on the max-norm, trace-norm and Rademacher complexity that will be used repeatedly later.

2.1 Notation

For any positive integer d, we use $[d]$ to denote the set of integers $\{1,2,...,d\}$. For any pair of real numbers a and b, set $a \vee b := \max(a,b)$ and $a \wedge b := \min(a,b)$. For a vector $u \in \mathbb{R}^d$ and $0 < p < \infty$, denote its ℓ_p-norm by $\|u\|_p = (\sum_{i=1}^d |u_i|^p)^{1/p}$. In particular, $\|u\|_\infty = \max_{i=1,...,d} |u_i|$ is the ℓ_∞-norm. For a matrix $M = (M_{k,l}) \in \mathbb{R}^{d_1 \times d_2}$, let $\|M\|_F = \sqrt{\sum_{k=1}^{d_1} \sum_{l=1}^{d_2} M_{k,l}^2}$ be the Frobenius norm and let $\|M\|_\infty = \max_{k,l} |M_{k,l}|$ denote the elementwise ℓ_∞-norm. Given two norms ℓ_p and ℓ_q on $\mathbb{R}^{d_1}$ and $\mathbb{R}^{d_2}$ respectively, the corresponding operator norm $\|\cdot\|_{p,q}$ of a matrix $M \in \mathbb{R}^{d_1 \times d_2}$ is defined by $\|M\|_{p,q} = \sup_{\|x\|_p=1} \|Mx\|_q$. It is easy to verify that $\|M\|_{p,q} = \|M^T\|_{q^*,p^*}$, where (p,p^*) and (q,q^*) are conjugate pairs, that is, $\frac{1}{p} + \frac{1}{p^*} = 1$ and $\frac{1}{q} + \frac{1}{q^*} = 1$. In particular, $\|M\| = \|M\|_{2,2}$ is the spectral norm and $\|M\|_{2,\infty} = \max_{k=1,...,d_1} \sqrt{\sum_{l=1}^{d_2} M_{k,l}^2}$ is the maximum row norm of M.

2.2 Max-Norm and Trace-Norm

For any matrix $M \in \mathbb{R}^{d_1 \times d_2}$, its *trace-norm* is defined to be the sum of the singular values of M (that is, the roots of the eigenvalues of MM^T), and can also equivalently written as

$$\|M\|_* = \inf\left\{ \sum_j |\sigma_j| : M = \sum_j \sigma_j u_j v_j^T,\, u_j \in \mathbb{R}^{d_1}, v_j \in \mathbb{R}^{d_2} \text{ satisfying } \|u_j\|_2 = \|v_j\|_2 = 1 \right\}.$$

Recall the definition (1) of the max-norm, the trace-norm can be analogously defined in terms of matrix factorization as

$$\|M\|_* = \min_{M=UV^T} \left\{ \|U\|_F \|V\|_F \right\} = \frac{1}{2} \min_{U,V:M=UV^T} \left(\|U\|_F^2 + \|V\|_F^2 \right).$$

Since the ℓ_1-norm of a vector is bounded by the product of its ℓ_2-norm and the number of non-zero coordinates, we have the following relationship between the trace-norm and Frobenius norm

$$\|M\|_F \le \|M\|_* \le \sqrt{\operatorname{rank}(M)} \cdot \|M\|_F.$$

By the elementary inequality $\|M_{m\times n}\|_F \le \sqrt{m}\|M_{m\times n}\|_{2,\infty}$, we see that

$$\frac{\|M\|_*}{\sqrt{d_1 d_2}} \le \|M\|_{\max}. \tag{2}$$

Furthermore, as was noticed in Lee et al. (2010), the max-norm, which is defined in (1), is comparable with a trace-norm more precisely in the following sense (Jameson, 1987):

$$\begin{aligned} &\|M\|_{\max} \\ &\quad \approx \inf\left\{ \sum_j |\sigma_j| : M = \sum_j \sigma_j u_j v_j^T,\, u_j \in \mathbb{R}^{d_1}, v_j \in \mathbb{R}^{d_2} \text{ satisfying } \|u_j\|_\infty = \|v_j\|_\infty = 1 \right\}, \end{aligned} \tag{3}$$

where the factor of equivalence is $K_G \in (1.67, 1.79)$, denoting the Grothendieck's constant. What may be more surprising is the following bounds for the max-norm, in connection with element-wise ℓ_∞-norm (Linial et al., 2007):

$$\|M\|_\infty \le \|M\|_{\max} \le \sqrt{\operatorname{rank}(M)} \cdot \|M\|_{1,\infty} \le \sqrt{\operatorname{rank}(M)} \cdot \|M\|_\infty. \tag{4}$$

2.3 Rademacher Complexity

Considering matrices as functions from index pairs to entry values, a technical tool used in our proof involves data-dependent estimates of the *Rademacher complexity* of the classes that consist of low trace-norm and low max-norm matrices. We refer to Bartlett and Mendelson (2002) for a detailed introduction of this concept.

Definition 1 *Let $\mathcal{P}$ be a probability distribution on a set $\mathcal{X}$. Suppose that $X_1, \ldots, X_n$ are independent samples drawn from $\mathcal{X}$ according to $\mathcal{P}$, and set $S = \{X_1, \ldots, X_n\}$. For a class $\mathcal{F}$ of functions mapping from $\mathcal{X}$ to $\mathbb{R}$, its empirical Rademacher complexity over the sample S is defined by*

$$\widehat{R}_S(\mathcal{F}) = \frac{2}{|S|} \mathbb{E}_\varepsilon \Big[\sup_{f \in \mathcal{F}} \Big| \sum_{i=1}^n \varepsilon_i f(X_i) \Big| \Big],$$

where $\varepsilon = (\varepsilon_1, ..., \varepsilon_n)$ *is a Rademacher sequence. The Rademacher complexity with respect to the distribution* $\mathcal{P}$ *is the expectation, over a sample* S *of* $|S|$ *points drawn i.i.d. according to* $\mathcal{P}$*, denoted by*

$$R_{|S|}(\mathcal{F}) = \mathbb{E}_{S\sim\mathcal{P}}[\widehat{R}_S(\mathcal{F})].$$

The following properties regarding $\widehat{R}_S(\mathcal{F})$ are useful.

Proposition 2 *We have*

1. *If* $\mathcal{F} \subseteq \mathcal{G}$, $\widehat{R}_S(\mathcal{F}) \leq \widehat{R}_S(\mathcal{G})$.
2. $\widehat{R}_S(\mathcal{F}) = \widehat{R}_S(\mathrm{conv}(\mathcal{F})) = \widehat{R}_S(\mathrm{absconv}(\mathcal{F}))$*, where* $\mathrm{conv}(\mathcal{F})$ *is the class of convex combinations of functions from* $\mathcal{F}$*, and* $\mathrm{absconv}(\mathcal{F})$ *denotes the absolutely convex hull of* $\mathcal{F}$*, that is, the class of convex combinations of functions from* $\mathcal{F}$ *and* $-\mathcal{F}$.
3. *For every* $c \in \mathbb{R}$, $\widehat{R}_S(c\mathcal{F}) = |c|\widehat{R}_S(\mathcal{F})$*, where* $c\mathcal{F} \equiv \{cf : f \in \mathcal{F}\}$.

In particular, we are interested in calculating the Rademacher complexities of the trace-norm and max-norm balls. To this end, define for any radius $R > 0$ that

$$\begin{aligned}\mathbb{B}_*(R) &:= \left\{M \in \mathbb{R}^{d_1\times d_2} : \|M\|_* \leq R\right\} \quad \text{and}\\ \mathbb{B}_{\max}(R) &:= \left\{M \in \mathbb{R}^{d_1\times d_2} : \|M\|_{\max} \leq R\right\}.\end{aligned}$$

First, recall that any matrix with unit trace-norm is a convex combination of unit-norm rank-one matrices, and thus

$$\mathbb{B}_*(1) = \mathrm{conv}(\mathcal{M}_1), \quad \text{where } \mathcal{M}_1 := \left\{uv^T : u \in \mathbb{R}^{d_1}, v \in \mathbb{R}^{d_2}, \|u\|_2 = \|v\|_2 = 1\right\}. \tag{5}$$

Then $\widehat{R}_S(\mathbb{B}_*(1)) = \widehat{R}_S(\mathcal{M}_1)$. A sharp bound on the worst-case Rademacher complexity, defined as the supremum of $\widehat{R}_S(\cdot)$ over all sample sets S with size $|S| = n$, is $\frac{2}{\sqrt{n}}$ (See, expression (4) on page 551, Srebro and Shraibman, 2005). This bound, unfortunately, is barely useful in developing generalization error bounds. However, when the index pairs of a sample S are drawn uniformly at random from $[d_1] \times [d_2]$ (with replacement), Srebro and Shraibman (2005) showed that the *expected* Rademacher complexity is low, and Foygel and Srebro (2011) have improved this result by reducing the logarithmic factor. In particular, they proved that for a sample size $n \geq d = d_1 + d_2$,

$$\mathbb{E}_{S\sim\mathrm{unif},|S|=n}\left[\widehat{R}_S(\mathbb{B}_*(1))\right] \leq \frac{K}{\sqrt{d_1 d_2}}\sqrt{\frac{d\log(d)}{n}},$$

where $K > 0$ denotes a universal constant.

The unit max-norm ball, on the other hand, can be approximately characterized as a convex hull. Due to the Grothendieck's inequality, it was shown in Srebro and Shraibman (2005) that

$$\mathrm{conv}(\mathcal{M}_\pm) \subset \mathbb{B}_{\max}(1) \subset K_G \cdot \mathrm{conv}(\mathcal{M}_\pm),$$

where $\mathcal{M}_\pm := \{M \in \{\pm 1\}^{d_1\times d_2} : \mathrm{rank}(M) = 1\}$ is the class of rank-one sign matrices, and $K_G \in (1.67, 1.79)$ is the Grothendieck's constant. It is easy to see that $\mathcal{M}_\pm$ is a finite class with cardinality

$|\mathcal{M}_{\pm}| = 2^{d-1}$, $d = d_1 + d_2$. For any $d_1, d_2 > 2$ and any sample of size $2 < |S| \leq d_1 d_2$, the empirical Rademacher complexity of the unit max-norm ball is bounded by

$$\widehat{R}_S\big(\mathbb{B}_{\max}(1)\big) \leq 12\sqrt{\frac{d}{|S|}}. \tag{6}$$

In other words, $\sup_{S:|S|=n} \widehat{R}_S(\mathbb{B}_{\max}(1)) \leq 12\sqrt{\frac{d}{n}}$.

2.4 Discrepancy

In order to get both upper and lower prediction error bounds on the weighted squared Frobenius norm between the proposed estimator, given by (13) below, and the target matrix described via model (9), we will need the following two concepts of discrepancies between matrices as well as their connections. In particular, we will focus on element-wise notion of discrepancy between two $d_1 \times d_2$ matrices P and Q.

First, for two matrices $P, Q : [d_1] \times [d_2] \to [0,1]^{d_1 \times d_2}$, their Hellinger distance is given by

$$d_H^2(P;Q) = \frac{1}{d_1 d_2} \sum_{(k,l)} d_H^2(P_{k,l}; Q_{k,l}),$$

where $d_H^2(p;q) = (\sqrt{p} - \sqrt{q})^2 + (\sqrt{1-p} - \sqrt{1-q})^2$ for $p, q \in [0,1]$. Next, the Kullback-Leibler divergence between two matrices $P, Q : [d_1] \times [d_2] \to [0,1]^{d_1 \times d_2}$ is defined by

$$\mathbb{K}(P\|Q) = \frac{1}{d_1 d_2} \sum_{(k,l)} K(P_{k,l}\|Q_{k,l}),$$

where $K(p\|q) = p\log(\frac{p}{q}) + (1-p)\log(\frac{1-p}{1-q})$, for $p, q \in [0,1]$. Note that $\mathbb{K}(P\|Q)$ is not a distance; it is sufficient to observe that it is not symmetric.

The relationship between the two "distances" is as follows. For any two scalars $p, q \in [0,1]$, we have

$$d_H^2(p;q) \leq K(p\|q), \tag{7}$$

which in turn implies that, for any two matrices $P, Q : [d_1] \times [d_2] \to [0,1]^{d_1 \times d_2}$,

$$d_H^2(P;Q) \leq \mathbb{K}(P\|Q). \tag{8}$$

The proof of (7) is based on the Jensen's inequality and an elementary inequality that $1 - x \leq -\log x$ for any $x > 0$.

3. Max-Norm Constrained Maximum Likelihood Estimate

In this section, we introduce the max-norm constrained maximum likelihood estimation procedure for 1-bit matrix completion and investigates the theoretical properties of the estimator. The results are also compared with other results in the literature.

3.1 Observation Model

We consider 1-bit matrix completion under a general random sampling model. The unknown low-rank matrix $M^* \in \mathbb{R}^{d_1 \times d_2}$ is the object of interest. Instead of observing noisy entries $M^*_{i,j} + Z_{i,j}$ directly in *unquantized* matrix completion, now we only observe with error the sign of a random subset of the entries of M^*. More specifically, assume that a random sample

$$S = \big\{(i_1, j_1), (i_2, j_2), ..., (i_n, j_n)\big\} \subseteq \big([d_1] \times [d_2]\big)^n$$

of the index set is drawn i.i.d. with replacement according to a general sampling distribution $\Pi = \{\pi_{kl}\}$ on $[d_1] \times [d_2]$. That is, $\mathbb{P}\{(i_t, j_t) = (k,l)\} = \pi_{kl}$, for all t and (k,l). Suppose that a (random) subset S of size $|S| = n$ of entries of a sign matrix Y is observed. The dependence of Y on the underlying matrix M^* is as follows:

$$Y_{i,j} = \begin{cases} +1, & \text{if } M^*_{i,j} + Z_{i,j} \geq 0, \\ -1, & \text{if } M^*_{i,j} + Z_{i,j} < 0, \end{cases} \tag{9}$$

where $Z = (Z_{i,j}) \in \mathbb{R}^{d_1 \times d_2}$ is a matrix consisting of i.i.d. noise variables. Let $F(\cdot)$ be the cumulative distribution function of $-Z_{1,1}$, then the above model can be recast as

$$Y_{i,j} = \begin{cases} +1, & \text{with probability } F(M^*_{i,j}), \\ -1, & \text{with probability } 1 - F(M^*_{i,j}), \end{cases} \tag{10}$$

and we observe noisy entries $\{Y_{i_t,j_t}\}_{t=1}^n$ indexed by S. More generally, we consider the model (10) with an arbitrary differentiable function $F : \mathbb{R} \to [0,1]$. Particular assumptions on F will be discussed below.

Instead of assuming the uniform sampling distribution as in Davenport et al. (2012), here we allow a general sampling distribution $\Pi = \{\pi_{kl}\}$, satisfying $\sum_{(k,l) \in [d_1] \times [d_2]} \pi_{kl} = 1$, according to which we make n independent random choices of entries. The drawback of the setting is that, with fairly high probability, some entries will be sampled multiple times. Intuitively it would be more practical to assume that entries are sampled without replacement, or equivalently, to sample n of the $d_1 d_2$ binary entries observed with noise without replacing. Due to the requirement that the drawn entries be distinct, the n samples are not independent. This dependence structure turns out to impede the technical analysis of the learning guarantees. To avoid this complication, we will use the i.i.d. approach as a proxy for sampling without replacement throughout this paper. As has been noted in Gross and Nesme (2010) and Foygel and Srebro (2011), between sampling with and without replacement both in a uniform sense, that is, making n independent uniform choices of entries versus choosing a set S of entries uniformly at random over all subsets that consist of exactly n entries, the latter can be theoretically as good as the former. See Section 7.4 below for more details.

Next we list three natural choices for F, or equivalently, for the distribution of $\{Z_{i,j}\}$.

3.1.1 EXAMPLES

1. (Logistic regression/Logistic noise): The logistic regression model is described by (10) with

$$F(x) = \frac{e^x}{1 + e^x},$$

and equivalently by (9) with $Z_{i,j}$ i.i.d. following the standard logistic distribution.

2. (Probit regression/Gaussian noise): The probit regression model is described by (10) with

$$F(x) = \Phi\Big(\frac{x}{\sigma}\Big),$$

where Φ denotes the cumulative distribution function of $N(0,1)$, and equivalently by (9) with $Z_{i,j}$ i.i.d. following $N(0,\sigma^2)$.

3. (Laplacian noise): Another interesting case is that the noise $Z_{i,j}$ are i.i.d. drawn from a Laplacian distribution Laplace$(0,b)$, with

$$F(x) = \begin{cases} \frac{1}{2}\exp(x/b), & \text{if } x < 0, \\ 1 - \frac{1}{2}\exp(-x/b), & \text{if } x \geq 0, \end{cases}$$

where $b > 0$ is the scale parameter.

Davenport et al. (2012) have focused on approximately low-rank matrices recovery by considering the following class of matrices

$$K_*(\alpha, r) = \Big\{ M \in \mathbb{R}^{d_1 \times d_2} : \|M\|_\infty \leq \alpha,\ \frac{\|M\|_*}{\sqrt{d_1 d_2}} \leq \alpha\sqrt{r} \Big\}, \tag{11}$$

where $1 \leq r \leq \min(d_1, d_2)$ and $\alpha > 0$ is a free parameter to be determined. Clearly, any matrix M with rank at most r satisfying $\|M\|_\infty \leq \alpha$ belongs to $K_*(\alpha, r)$. Alternatively, using max-norm as a convex relaxation for the rank, we consider recovery of matrices with ℓ_∞-norm and max-norm constraints defined by

$$K_{\max}(\alpha, R) := \Big\{ M \in \mathbb{R}^{d_1 \times d_2} : \|M\|_\infty \leq \alpha,\ \|M\|_{\max} \leq R \Big\}. \tag{12}$$

Here both $\alpha > 0$ and $R > 0$ are free parameters to be determined. If M^* is of rank at most r and $\|M^*\|_\infty \leq \alpha$, then by (2) and (4) we have $M^* \in \mathbb{B}_{\max}(\alpha\sqrt{r})$ and hence

$$M^* \in K_{\max}(\alpha, \alpha\sqrt{r}) \subset K_*(\alpha, r).$$

3.2 Max-norm Constrained Maximum Likelihood Estimate

Now, given a collection of observations $Y_S = \{Y_{i_t,j_t}\}_{t=1}^n$ from the observation model (10), the negative log-likelihood function can be written as

$$\ell_S(M; Y) = \sum_{t=1}^{n} \left[\mathbf{1}_{\{Y_{i_t,j_t}=1\}} \log\Big(\frac{1}{F(M_{i_t,j_t})}\Big) + \mathbf{1}_{\{Y_{i_t,j_t}=-1\}} \log\Big(\frac{1}{1 - F(M_{i_t,j_t})}\Big) \right].$$

Then we consider estimating the unknown $M^* \in K_{\max}(\alpha, R)$ by maximizing the empirical likelihood function subject to a max-norm constraint:

$$\widehat{M}_{\max} = \underset{M \in K_{\max}(\alpha,R)}{\arg\min}\ \ell_S(M; Y). \tag{13}$$

The optimization procedure requires that all the entries of M_0 are bounded in absolute value by a pre-defined constant α. This condition is reasonable while also critical in approximate low-rank

matrix recovery problems by controlling the *spikiness* of the solution. Indeed, the measure of the "spikiness" of matrices is much less restrictive than the incoherence conditions imposed in exact low-rank matrix recovery. See, for example, Koltchinskii et al. (2011), Negahban and Wainwright (2012), Klopp (2012) and Cai and Zhou (2013).

As has been noted in Srebro et al. (2005), a large gap between the max-complexity (related to max-norm) and the dimensional-complexity (related to rank) is possible only when the underlying low-rank matrix has entries of vastly varying magnitudes. Also, in view of (3), the max-norm promotes low-rank decomposition with factors in ℓ_∞ (ℓ_2 for the trace-norm). Motivated by these features, max-norm regularization is expected to be reasonably effective for uniformly bounded data.

When the noise distribution is log-concave so that the log-likelihood is a concave function, the max-norm constrained minimization problem (13) is a convex program and we recommend a fast and efficient algorithm developed in Lee et al. (2010) for solving large-scale optimization problems that incorporate the max-norm. We will show in Section 4 that the convex optimization problem (13) can indeed be implemented in polynomial time as a function of the sample size n and the matrix dimensions d_1 and d_2.

3.3 Upper Bounds

To establish an upper bound on the prediction error of estimator $\widehat{M}_{\max}$ given by (13), we need the following assumption on the unknown matrix M^* as well as the regularity conditions on the function F in (10).

3.3.1 CONDITION U

Assume that there exist positive constants R and α such that

(U1) $M^* \in K_{\max}(\alpha, R)$;

(U2) F and F' are non-zero in $[-\alpha, \alpha]$, and

(U3) both

$$L_\alpha := \sup_{|x|\le\alpha} \frac{|F'(x)|}{F(x)(1-F(x))}, \quad \text{and} \quad \beta_\alpha := \sup_{|x|\le\alpha} \frac{F(x)(1-F(x))}{(F'(x))^2} \tag{14}$$

are finite.

In particular, under condition (U2), the quantity

$$U_\alpha := \sup_{|x|\le\alpha} \log\left(\frac{1}{F(x)(1-F(x))}\right),$$

is well-defined. As prototypical examples, we specify below the quantities L_α, β_α and U_α in the cases of Logistic, Gaussian and Laplacian noise:

1. (Logistic regression/Logistic noise): For $F(x) = e^x/(1+e^x)$, we have

$$L_\alpha \equiv 1, \ \ \beta_\alpha = \frac{(1+e^\alpha)^2}{e^\alpha} \quad \text{and} \quad U_\alpha = 2\log(e^{\alpha/2} + e^{-\alpha/2}).$$

2. (Probit regression/Gaussian noise): For $F(x) = \Phi(x/\sigma)$, straightforward calculations show that
$$L_\alpha \le \frac{4}{\sigma}\Big(\frac{\alpha}{\sigma}+1\Big),\ \ \beta_\alpha \le \pi\sigma^2 \exp\{\alpha^2/(2\sigma^2)\} \quad \text{and} \quad U_\alpha \le \Big(\frac{\alpha}{\sigma}+1\Big)^2. \tag{15}$$

3. (Laplacian noise): For a Laplace$(0,b)$ distribution function, we have
$$L_\alpha = \frac{2}{b},\ \ \beta_\alpha = b\big(2\exp(\alpha/b)-1\big) \quad \text{and} \quad U_\alpha \le 2\Big(\frac{\alpha}{b}+\log 2\Big).$$

Now we are ready to state our main results concerning the recovery of an approximately low-rank matrix M^* using the max-norm constrained maximum likelihood estimate. We write hereafter $d = d_1 + d_2$ for brevity.

Theorem 3 *Suppose that Condition U holds and assume that the training set S follows a general weighted sampling model according to the distribution* Π. *Then there exists an absolute constant C such that, for a sample size* $2 < n \le d_1 d_2$ *and for any* $\delta > 0$, *the minimizer* $\widehat{M}_{\max}$ *of the optimization program* (13) *satisfies*
$$\|\widehat{M}_{\max} - M^*\|_\Pi^2 = \sum_{k=1}^{d_1}\sum_{l=1}^{d_2} \pi_{kl}\{\widehat{M}_{max} - M^*\}_{k,l}^2 \le C\beta_\alpha\left\{L_\alpha R\sqrt{\frac{d}{n}} + U_\alpha\sqrt{\frac{\log(4/\delta)}{n}}\right\}, \tag{16}$$
with probability at least $1-\delta$. *Here and below* $\|\cdot\|_\Pi$ *denotes the weighted Frobenius norm with respect to* Π, *that is,*
$$\|M\|_\Pi = \sqrt{\sum_{k=1}^{d_1}\sum_{l=1}^{d_2}\pi_{kl}M_{k,l}^2} \quad \textit{for all } M \in \mathbb{R}^{d_1\times d_2}.$$

Remark 4 (i) While using the trace-norm to study this general weighted sampling model, it is common to assume that each row and column is sampled with positive probability (Klopp, 2012; Negahban and Wainwright, 2012), though in some applications this assumption does not seem realistic. More precisely, assume that there exists a positive constant $\mu \ge 1$ such that
$$\pi_{kl} \ge \frac{1}{\mu d_1 d_2}, \quad \text{for all } (k,l) \in [d_1]\times[d_2]. \tag{17}$$
Then, under condition (17) and the conditions of Theorem 3,
$$\frac{1}{d_1 d_2}\|\widehat{M}_{\max} - M^*\|_F^2 \le C\mu\beta_\alpha\left\{L_\alpha R\sqrt{\frac{d}{n}} + U_\alpha\sqrt{\frac{\log(d)}{n}}\right\} \tag{18}$$
holds with probability at least $1-4/d$, where $C>0$ denotes an absolute constant.

(ii) Klopp (2012) studied the problem of standard matrix completion with noise, also in the case of general sampling distribution, using the trace-norm penalized approach. However, the Assumption 1 therein requires that the distribution π_{kl} over entries is bounded from above, which is quite restrictive especially in the Netflix problem. It is worth noticing that this upper bound condition on sampling distribution is not required in both results (16) and (18).

It is noteworthy that above results are directly comparable to those obtained in the case of approximately low-rank recovery from unquantized measurements, also using max-norm regularized approach (Cai and Zhou, 2013). Let $Z = (Z_{i,j})$ be a noise matrix consisting of i.i.d. $N(0,\sigma^2)$ entries for some $\sigma > 0$, and assume we have observations on a (random) subset $S = \{(i_1, j_1), ..., (i_n, j_n)\}$ of entries of $\widetilde{Y} = M^* + Z$. Cai and Zhou (2013) studied the unquantized problem under a general sampling model using max-norm as a convex relaxation for the rank. In particular, for the max-norm constrained least squares estimator

$$\widetilde{M}_{\max} = \underset{M \in K_{\max}(\alpha,R)}{\arg\min} \frac{1}{n} \sum_{t=1}^{n} (\widetilde{Y}_{i_t,j_t} - M^*_{i_t,j_t})^2,$$

for $K_{\max}(\alpha, R)$ as in (12), it was shown that for any $\delta \in (0,1)$ and a sample size $2 < n \le d_1 d_2$,

$$\|\widetilde{M}_{\max} - M^*\|_\Pi^2 \le C' \left\{ (\alpha \vee \sigma) R \sqrt{\frac{d}{n}} + \frac{\alpha^2 \log(2/\delta)}{n} \right\} \tag{19}$$

holds with probability greater than $1 - \exp(-d) - \delta$, where $C' > 0$ is a universal constant.

In 1-bit observations case when $Z_{i,j} \overset{i.i.d.}{\sim} N(0,\sigma^2)$, it is equivalent that the function F in model (10) is given by $F(\cdot) = \Phi(\cdot/\sigma)$. According to (15), we have

$$\|\widehat{M}_{\max} - M^*\|_\Pi^2 \le C \exp\left(\frac{\alpha^2}{2\sigma^2}\right) \left\{ (\alpha+\sigma) R \sqrt{\frac{d}{n}} + (\alpha+\sigma)^2 \sqrt{\frac{\log(4/\delta)}{n}} \right\} \tag{20}$$

holds with probability at least $1 - \delta$.

Comparing the upper bounds in (19) and (20) and note that $\alpha \vee \sigma \le \alpha + \sigma \le 2(\alpha \vee \sigma)$, we see that there is no essential loss of recovery accuracy by discretizing to binary measurements as long as $\frac{\alpha}{\sigma}$ is bounded by a constant (Davenport et al., 2012). On the other hand, as the signal-to-noise ratio $\frac{\alpha}{\sigma} \ge 1$ increases, the error bounds deteriorate significantly. In fact, the case $\alpha \gg \sigma$ essentially amounts to the noiseless setting, in which it is impossible to recover M^* based on any subset of the signs of its entries.

3.4 Information-Theoretic Lower Bounds

We now establish minimax lower bounds by using information-theoretic techniques. The lower bounds given in Theorem 5 below show that the rate attained by the max-norm constrained maximum likelihood estimator is optimal up to constant factors.

Theorem 5 *Assume that $F'(x)$ is decreasing and $\frac{F(x)(1-F(x))}{(F'(x))^2}$ is increasing for $x > 0$, and let S be any subset of $[d_1] \times [d_2]$ with cardinality n. Then, as long as the parameters (R, α) satisfy*

$$\max\left(2, \frac{4}{(d_1 \vee d_2)^{1/2}}\right) \le \frac{R}{\alpha} \le \frac{(d_1 \wedge d_2)^{1/2}}{2},$$

the minimax risk for estimating M over the parameter space $K_{\max}(\alpha, R)$ satisfies

$$\inf_{\widehat{M}} \max_{M \in K_{\max}(\alpha,R)} \left\{ \frac{1}{d_1 d_2} \mathbb{E} \|\widehat{M} - M\|_F^2 \right\} \ge \frac{1}{512} \min\left\{ \alpha^2, \frac{\sqrt{\beta_{\alpha/2}}}{2} R \sqrt{\frac{d}{n}} \right\}. \tag{21}$$

Remark 6 In fact, the lower bound (21) is a special case of the following general result, which will be proved in Sect. 7.2. Let $\gamma^* > 0$ be the solution of the following equation

$$\gamma^* = \min\left\{\frac{1}{2}, \frac{R^{1/2}}{\alpha}\left(\frac{\beta_{(1-\gamma^*)\alpha}}{32}\cdot\frac{d_1\vee d_2}{n}\right)^{1/4}\right\} \tag{22}$$

and assume that

$$\max\left(2, \frac{4}{(d_1\vee d_2)^{1/2}}\right) \le \frac{R}{\alpha} \le (d_1\wedge d_2)^{1/2}\gamma^*. \tag{23}$$

Then the minimax risk for estimating M over the parameter space $K_{\max}(\alpha, R)$ satisfies

$$\inf_{\widehat{M}}\max_{M\in K_{\max}(\alpha,R)}\left\{\frac{1}{d_1d_2}\mathbb{E}\|\widehat{M}-M\|_F^2\right\} \ge \frac{1}{512}\min\left\{\alpha^2, \frac{\sqrt{\beta_{(1-\gamma^*)\alpha}}}{2}R\sqrt{\frac{d}{n}}\right\}. \tag{24}$$

To see the existence of γ^* defined above, setting

$$h(\gamma) = \gamma \quad \text{and} \quad g(\gamma) = \min\left\{\frac{1}{2}, \frac{R^{1/2}}{\alpha}\left(\frac{\beta_{(1-\gamma)\alpha}}{32}\cdot\frac{d_1\vee d_2}{n}\right)^{1/4}\right\},$$

then it is easy to see that $h(\gamma)$ is strictly increasing and $g(\gamma)$ is decreasing for $\gamma\in(0,1)$ with $h(0)=0$ and $g(0)>0$. Therefore, equation (22) has a unique solution $\gamma^*\in(0,\frac{1}{2}]$ so that $h(\gamma^*)=g(\gamma^*)$.

Assume that μ and α are bounded above by universal constants and let the function F be fixed, so that both L_α and β_α in (14) are bounded. Also notice that $\beta_{(1-\gamma^*)\alpha}\ge\beta_{\alpha/2}$ since $\gamma^*\le 1/2$. Then comparing the lower bound (24) with the upper bound (18) shows that if the sample size $n\ge\frac{R^2\beta_{\alpha/2}}{4\alpha^4}(d_1+d_2)$, the optimal rate of convergence is $R\sqrt{\frac{d_1+d_2}{n}}$:

$$\inf_{\widehat{M}}\sup_{M\in K_{\max}(\alpha,R)}\frac{1}{d_1d_2}\mathbb{E}\|\widehat{M}-M\|_F^2 \asymp R\sqrt{\frac{d_1+d_2}{n}},$$

and the max-norm constrained maximum likelihood estimate (13) is rate-optimal. If the target matrix M^* is known to have rank at most r, we can take $R=\alpha\sqrt{r}$, such that the requirement here on the sample size $n\ge\frac{\beta_{\alpha/2}}{4\alpha^2}r(d_1+d_2)$ is weak and the optimal rate of convergence becomes $\alpha\sqrt{\frac{r(d_1+d_2)}{n}}$.

3.5 Comparison to Prior Work

In this paper, we study a matrix completion model proposed in Davenport et al. (2012), in which it is assumed that a binary matrix is observed at random from a distribution parameterized by an unknown matrix which is (approximately) low-rank. It is noteworthy that some earlier papers on collaborative filtering or matrix completion, including Srebro et al. (2005) and references therein, also dealt with binary observations that are assumed to be noisy versions of the underlying matrix, in Logistic or Bernoulli conditional model. The goal there is to predict directly the quantized values, or equivalently, to reconstruct the sign matrix, instead of the underlying real values, therefore the non-identifiability issue could be avoided.

We next turn to a detailed comparison of our results for 1-bit matrix completion to those obtained in Davenport et al. (2012), also for approximately low-rank matrices. Using the trace-norm as

a proxy to rank, Davenport et al. (2012) have studied 1-bit matrix completion under the *uniform sampling distribution* over the parameter space $K_*(\alpha, r)$ as given in (11), for some $\alpha > 0$ and $r \leq \min\{d_1, d_2\}$ is a positive integer. To recover the unknown $M^* \in K_*(\alpha, r)$, given a collection of observations Y_S where S follows a Bernoulli model, that is, every entry $(k,l) \in [d_1] \times [d_2]$ is observed independently with equal probability $\frac{n}{d_1 d_2}$, they propose the following trace-norm constrained MLE

$$\widehat{M}_{\text{tr}} = \underset{M \in K_*(\alpha, r)}{\arg\min} \ \ell_S(M; Y)$$

and prove that for a sample size $n \geq d \log(d)$, $d = d_1 + d_2$, with high probability,

$$\frac{1}{d_1 d_2} \|\widehat{M}_{\text{tr}} - M^*\|_F^2 \lesssim \beta_\alpha L_\alpha \alpha \sqrt{\frac{rd}{n}}. \tag{25}$$

Comparing to (18) with $R = \alpha\sqrt{r}$, it is easy to see that under the uniform sampling model, the error bounds in (rescaled) Frobenius norm for the two estimates $\widehat{M}_{\max}$ and $\widehat{M}_{\text{tr}}$ are of the same order. Moreover, Theorem 3 in Davenport et al. (2012) and Theorem 5, respectively, provide lower bounds showing that both $\widehat{M}_{\text{tr}}$ and $\widehat{M}_{\max}$ achieve the minimax rate of convergence for recovering approximately low-rank matrices over the parameter spaces $K_*(\alpha, r)$ and $K_{\max}(\alpha, R)$ respectively.

As mentioned in the introduction, the uniform sampling distribution assumption is restrictive and not valid in many applications including the well-known Netflix problem. When the sampling distribution is non-uniform, it was shown in Salakhutdinov and Srebro (2010) that the standard trace-norm regularized method might fail, specifically in the setting where the row and column marginal distributions are such that certain rows or columns are sampled with very high probabilities. Moreover, it was proposed to use a weighted variant of the trace-norm, which incorporates the knowledge of the true sampling distribution in its construction, and showed experimentally that this variant indeed leads to superior performance. Using this weighted trace-norm, Negahban and Wainwright (2012) provided theoretical guarantees on approximate low-rank matrix completion in general sampling case while assuming that each row and column is sampled with positive probability (see condition (17)). In addition, requiring that the probabilities to observe an element from any row or column are of order $O((d_1 \wedge d_2)^{-1})$, Klopp (2012) analyzed the performance of the trace-norm penalized estimators, and provided near-optimal (up to a logarithmic factor) bounds which are similar to the bounds in this paper.

Next we provide an analysis of the performance of the weighted trace-norm in 1-bit matrix completion. Given the knowledge of the true sampling distribution, we establish an upper bound on the error in recovering M^*, which comparing to (25), includes an additional $\log^{1/2}(d)$ factor. We do not rule out the possibility that this logarithmic factor might be an artifact of the technical tools used in proof described below. The proof in Davenport et al. (2012) for the trace-norm regularization in uniform sampling case may also be extended to the weighted trace-norm method under the general sampling model, by using the matrix Bernstein inequality instead of Seginer's theorem. The extra logarithmic factor, however, is still inevitable based on this argument. We will not pursue the details in this paper.

Given a sampling distribution $\Pi = \{\pi_{kl}\}$ on $[d_1] \times [d_2]$, define its row- and column-marginals as

$$\pi_{k\cdot} = \sum_{l=1}^{d_2} \pi_{kl} \quad \text{and} \quad \pi_{\cdot l} = \sum_{k=1}^{d_1} \pi_{kl},$$

respectively. Under the condition (17), we have

$$\pi_{k\cdot} \geq \frac{1}{\mu d_1}, \;\; \pi_{\cdot l} \geq \frac{1}{\mu d_2}, \quad \text{for all } (k,l) \in [d_1] \times [d_2]. \tag{26}$$

As in Salakhutdinov and Srebro (2010), consider the following weighted trace-norm with respect to the distribution Π:

$$\|M\|_{w,*} := \|M_w\|_* = \left\|\mathrm{diag}(\sqrt{\pi_{1\cdot}}, ..., \sqrt{\pi_{d_1\cdot}}) \cdot M \cdot \mathrm{diag}(\sqrt{\pi_{\cdot 1}}, ..., \sqrt{\pi_{\cdot d_2}})\right\|_*, \tag{27}$$

where $(M_w)_{k,l} := \sqrt{\pi_{k\cdot}\pi_{\cdot l}} M_{k,l}$. Notice that if M has rank at most r and $\|M\|_\infty \leq \alpha$, then

$$\|M\|_{w,*} \leq \sqrt{r}\|M\|_F = \sqrt{r}\left(\sum_{k=1}^{d_1}\sum_{l=1}^{d_2} \pi_{k\cdot}\pi_{\cdot l} M_{k,l}^2\right)^{1/2} \leq \alpha\sqrt{r}.$$

Analogously to the previous studied class $K_*(\alpha, r)$, as given in (11), containing the low trace-norm matrices, define

$$K_{\Pi,*} \equiv K_{\Pi,*}(r,\alpha) = \left\{M \in \mathbb{R}^{d_1 \times d_2} : \|M\|_{w,*} \leq \alpha\sqrt{r}, \|M\|_\infty \leq \alpha\right\}$$

and consider estimating the unknown $M^* \in K_{\Pi,*}$ by solving the following optimization problem:

$$\widehat{M}_{w,tr} = \underset{M \in K_{\Pi,*}}{\arg\min}\, \ell_S(M;Y). \tag{28}$$

The following theorem states that the weighted trace-norm regularized approach can be nearly as good as the max-norm regularized estimator (up to logarithmic and constant factors), under a general sampling distribution that is not too far from uniform. The theoretical performance of the weighted trace-norm is first studied by Foygel et al. (2011) in the standard matrix completion problems under arbitrary sampling distributions.

Theorem 7 *Suppose that Condition U holds but with $M^* \in K_{\Pi,*}$, and assume that the training set S follows a general weighted sampling model according to the distribution Π satisfying (17). Then there exists an absolute constant $C > 0$ such that, for a sample size $n \geq \mu \min\{d_1, d_2\}\log(d)$ and any $\delta > 0$, the minimizer $\widehat{M}_{w,tr}$ of the optimization program (28) satisfies*

$$\|\widehat{M}_{w,tr} - M^*\|_\Pi^2 \leq C\beta_\alpha\left\{L_\alpha\alpha\sqrt{\frac{\mu r d\log(d)}{n}} + U_\alpha\sqrt{\frac{\log(4/\delta)}{n}}\right\}, \tag{29}$$

with probability at least $1-\delta$.

Since the construction of weighted trace-norm $\|\cdot\|_{w,*}$ highly depends on the underlying sampling distribution which is typically unknown in practice, the constraint $M^* \in K_{\Pi,*}$ seems to be artificial. The max-norm constrained approach, on the contrary, does not require the knowledge of the exact sampling distribution and the error bound in weighted Frobenius norm, as shown in (16), holds even without prior assumption on Π, for example, condition (17). Moreover, to ensure that the weighted trace-norm regularized method performs well, it is necessary that the marginals are not too small or equivalently that

$$\mu = \max\left\{\frac{1}{d_1\pi_{k\cdot}} \vee \frac{1}{d_2\pi_{\cdot l}} : k,l \in [d_1] \times [d_2]\right\}$$

is not too large. Otherwise, both the error bounds in (29) and the sample complexity $\mu\min\{d_1,d_2\}\log(d)$ would grow larger with μ when the marginals were far from uniform. We conjecture that the factor μ would also appear in the results that are extended from those in Davenport et al. (2012). As evident in Theorem 3, using the max-norm based regularization does not lead to a deterioration in either the error bounds or the sample complexity when the sampling distribution was far from uniform.

To clarify the major difference between the principles behind (25) and (29), we remark that one of the key technical tools used in Davenport et al. (2012) is a bound of Seginer (2000) on the spectral norm of a random matrix with i.i.d. zero mean entries (corresponding to the uniform sampling distribution), that is, for any $h \leq 2\log(\max\{d_1,d_2\})$,

$$\mathbb{E}[\|A\|^h] \leq K^h\Big(\mathbb{E}\Big[\max_{k=1,\ldots,d_1}\|a_{k\cdot}\|_2^h\Big]+\mathbb{E}\Big[\max_{j=1,\ldots,d_2}\|a_{\cdot l}\|_2^h\Big]\Big),$$

where $a_{k\cdot}$ (resp. $a_{\cdot l}$) denote the rows (resp. columns) of A and K is a universal constant. Under the non-uniform sampling model, we will deal with a matrix with independent entries that are not necessarily identically distributed, to which case an alternative result of Latala (2005) can be applied, that is,

$$\mathbb{E}[\|A\|] \leq K'\Big(\max_{k=1,\ldots,d_1}\mathbb{E}\|a_{k\cdot}\|_2+\max_{j=1,\ldots,d_2}\mathbb{E}\|a_{\cdot l}\|_2+\Big(\sum_{k,l}\mathbb{E}a_{kl}^4\Big)^{1/4}\Big]\Big),$$

or instead, resorting to the matrix Bernstein inequality. Using either inequality would thus bring an additional logarithmic factor, appeared in (29).

It is also worth noticing that though the sampling distribution is not known exactly in practice, its empirical analogues are expected to be stable enough as an alternative. According to Foygel et al. (2011), given a random sample $S=\{(i_t,j_t)\}_{t=1}^n$, consider the empirical marginals

$$\widehat{\pi}^r(i)=\frac{\#\{t:i_t=i\}}{n},\ \ \widehat{\pi}^c(j)=\frac{\#\{t:j_t=j\}}{n}\quad\text{and}\quad\widehat{\pi}_{ij}=\widehat{\pi}^r(i)\widehat{\pi}^c(j),$$

as well as the smoothed empirical marginals

$$\check{\pi}^r(i)=\frac{1}{2}(\widehat{\pi}^r(i)+1/d_1),\ \ \widehat{\pi}^c(j)=\frac{1}{2}(\widehat{\pi}^c(j)+1/d_2)\quad\text{and}\quad\check{\pi}_{ij}=\check{\pi}^r(i)\check{\pi}^c(j).$$

The smoothed empirically-weighted trace-norm $\|\cdot\|_{\check{w},*}$ can be defined in the same spirit as in the definition (27) of weighted trace-norm, only with $\{\pi_{ij}\}$ replaced by $\{\check{\pi}_{ij}\}$. Then the unknown matrix can be estimated via regularization on the $\check{\pi}$-weighted trace-norm, that is,

$$\check{M}_{\check{w},tr}=\arg\min\big\{\ell_S(M;Y):\|M\|_\infty\leq\alpha,\ \|M\|_{\check{w},*}\leq\alpha\sqrt{r}\big\}.$$

Adopting Theorem 4 in Foygel et al. (2011) to the current 1-bit problem will lead to a learning guarantee similar to (29).

4. Computational Algorithm

Problems of the form (13) can now be solved using a variety of algorithms, including interior point method (Srebro et al., 2005), Frank-Wolfe-type algorithm (Jaggi, 2013) and projected gradient

method (Lee et al., 2010). The first two are convex methods with guaranteed convergence rates to the global optimum, though can be slow in practice and might not scale to matrices with hundreds of rows or columns. We describe in this section a simple first order method due to Lee et al. (2010), which is a special case of a projected gradient algorithm for solving large-scale convex programs involving the max-norm. This method is non-convex, but as long as the size of the problem is large enough, it is guaranteed that each local minimum is also a global optimum, due to Burer and Monteiro (2003).

We start from rewriting the original problem as an optimization over factorizations of a matrix $M \in \mathbb{R}^{d_1 \times d_2}$ into two terms $M = UV^T$, where $U \in \mathbb{R}^{d_1 \times k}$ and $V \in \mathbb{R}^{d_2 \times k}$ for some $1 \leq k \leq d = d_1 + d_2$. More specifically, for any $1 \leq k \leq d$ fixed, define

$$\mathcal{M}_k(R) := \Big\{ UV^T : U \in \mathbb{R}^{d_1 \times k}, V \in \mathbb{R}^{d_2 \times k}, \max\{\|U\|_{2,\infty}^2, \|V\|_{2,\infty}^2\} \leq R \Big\}.$$

Then the global optimum of (13) is equal to that of

$$\begin{array}{ll} \text{minimize} & \ell(M;Y) \\ \text{subject to} & M \in \mathcal{M}_k(R), \ \ \|M\|_\infty \leq \alpha. \end{array} \tag{30}$$

Here we write $\ell(M;Y) = \frac{1}{|S|}\ell_S(M;Y)$ for brevity. This problem is non-convex, come with no guaranteed convergence rates to the global optimum. A surprising fact is that when $k \geq 1$ is large enough, this problem has no local minimum (Burer and Monteiro, 2003). Notice that $\ell(\cdot;Y)$ is differentiable with respect to the first argument, then (30) can be solved iteratively via the following updates:

$$\begin{bmatrix} U(\tau) \\ V(\tau) \end{bmatrix} = \begin{bmatrix} U^t - \frac{\tau}{\sqrt{t}} \cdot \nabla f(U^t(V^t)^T;Y)V^t \\ V^t - \frac{\tau}{\sqrt{t}} \cdot \nabla f(U^t(V^t)^T;Y)^T U^t \end{bmatrix},$$

where $\tau > 0$ is a stepsize parameter and $t = 0, 1, 2, \ldots$. Next, we project $(U(\tau), V(\tau))$ onto $\mathcal{M}_k(R)$ according to

$$\begin{bmatrix} \widetilde{U}^{t+1} \\ \widetilde{V}^{t+1} \end{bmatrix} = \mathcal{P}_R\left(\begin{bmatrix} U(\tau) \\ V(\tau) \end{bmatrix} \right).$$

This orthogonal projection can be computed by re-scaling the rows of the current iterate whose ℓ_2-norms exceed R so that their norms become exactly R, while rows with norms already less than R remain unchanged. If $\|\widetilde{U}^{t+1}(\widetilde{V}^{t+1})^T\|_\infty > \alpha$, we replace

$$\begin{bmatrix} \widetilde{U}^{t+1} \\ \widetilde{V}^{t+1} \end{bmatrix} \quad \text{with} \quad \frac{\sqrt{\alpha}}{\|\widetilde{U}^{t+1}(\widetilde{V}^{t+1})^T\|_\infty^{1/2}} \begin{bmatrix} \widetilde{U}^{t+1} \\ \widetilde{V}^{t+1} \end{bmatrix},$$

otherwise we keep it still. The resulting update is then denoted by (U^{t+1}, V^{t+1}).

It is important to note that the choice of k must be large enough, at least as big as the rank of M^*. Suppose that, before solving (13), we know that the target matrix M^* has rank at most r^*. Then it is best to solve (30) for $k = r^* + 1$ in the sense that, if we choose $k \leq r^*$, then (30) is not equivalent to (13), and if we take $k > r^* + 1$, then we would be solving a larger program than necessary. In practice, we do not know the exact value of r^* in advance. Nevertheless, motivated by Burer and Monteiro (2003), we suggest the following scheme to solve the problem which avoids solving (30) for $r \gg r^*$:

(1) Choose an initial small k and compute a local minimum (U,V) of (30), using above projected gradient method.

(2) Use an optimization technique to determine whether the injections $\widehat{U}$ of U into $\mathbb{R}^{d_1\times(k+1)}$ and $\widehat{V}$ of V into $\mathbb{R}^{d_2\times(k+1)}$ comprise a local minimum of (30) with the size increased to $k+1$.

(3) If $(\widehat{U},\widehat{V})$ is a local minimum, then we can take $M = UV^T$ as the final solution; otherwise compute a better local minimum $(\widetilde{U},\widetilde{V})$ of (30) with size $k+1$ and repeat step (2) with $(U,V) = (\widetilde{U},\widetilde{V})$ and $k = k+1$.

It was also suggested in Lee et al. (2010) that when dealing with extremely large data sets with S consisting of hundreds of millions of index pairs, one may consider using a stochastic gradient method based on the following decomposition for ℓ, that is,

$$\begin{aligned}\ell(UV^T;Y) &= \frac{1}{|S|}\sum_{(i,j)\in S} g(u_i^T v_j; Y_{i,j}) \quad \text{with}\\ g(t;y) &= \mathbf{1}_{\{y=1\}}\log\left(\frac{1}{F(t)}\right)+\mathbf{1}_{\{y=-1\}}\log\left(\frac{1}{1-F(t)}\right),\end{aligned}$$

where $S\subset[d_1]\times[d_2]$ is a training set of row-column indices, u_i and v_j denote the i-th row of U and j-th row of V, respectively. The stochastic gradient method says that at t-th iteration, we only need to pick one training pair (i_t,j_t) at random from S, then update $g(u_{i_t}^T v_{j_t}; Y_{i_t,j_t})$ via the previous procedure. More precisely, if $\|u_{i_t}\|_2^2 > R$, we project it back so that $\|u_{i_t}\|_2^2 = R$, otherwise we do not make any change (do the same for v_{j_t}). Next, if $|u_{i_t}^T v_{j_t}| > \alpha$, replace u_{i_t} and v_{i_t} with $\sqrt{\alpha}u_{i_t}/|u_{i_t}^T v_{j_t}|^{1/2}$ and $\sqrt{\alpha}v_{i_t}/|u_{i_t}^T v_{j_t}|^{1/2}$ respectively, otherwise we keep everything still. At the t-th iteration, we do not need to consider any other rows of U and V. This simple algorithm could be computationally as efficient as optimization with the trace-norm.

5. Numerical Results

In this section, we report the simulation results for low-rank matrix recovery based on 1-bit observations. In all cases presented below, we solved the convex program (30) by using our implementation in MATLAB of the projected gradient algorithm proposed in Section 4 for a wide range of values of the step-size parameter τ.

We first consider a rank-2, $d\times d$ target matrix M^* with eigenvalues $\{d/\sqrt{2}, d/\sqrt{2}, 0, ..., 0\}$, so that $\|M^*\|_F/d = 1$. We choose to work with the Gaussian conditional model under uniform sampling. Let Y_S be the noisy binary observations with $S = \{(i_1,j_1),...,(i_t,j_t)\}$, that is, for $(i,j)\in S$,

$$Y_{i,j} = \begin{cases} +1, & \text{with probability } \Phi(M^*_{i,j}/\sigma),\\ -1, & \text{with probability } 1-\Phi(M^*_{i,j}/\sigma),\end{cases}$$

and the objective function is given by

$$\ell_S(M;Y) = \frac{1}{|S|}\left\{\sum_{(i,j)\in\Omega^+}\log\left[\frac{1}{\Phi(M_{i,j}/\sigma)}\right]+\sum_{(i,j)\in\Omega^-}\log\left[\frac{1}{1-\Phi(M_{i,j}/\sigma)}\right]\right\},$$

where $\Omega^+ = \{(i,j)\in S : Y_{i,j} = 1\}$ and $\Omega^- = \{(i,j)\in S : Y_{i,j} = -1\}$. In Figure 1, averaging the results over 20 repetitions, we plot the squared Frobenius norm of the error (normalized by the

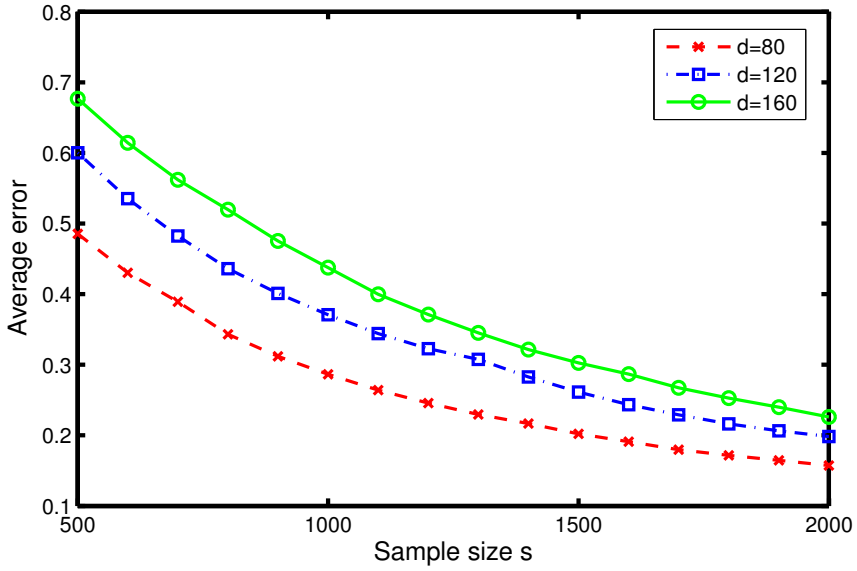


Figure 1: Plot of the average Frobenius error $\|\widehat{M} - M^*\|_F^2/d^2$ versus the sample size s for three different matrix sizes $d \in \{80, 120, 160\}$, all with rank $r = 2$.

dimension) $\|\widehat{M} - M^*\|_F^2/d^2$ versus a range of sample sizes $s = |S|$, with the noise level σ taken to be $\alpha/2$, for three different matrix sizes, $d \in \{80, 120, 160\}$. Naturally, in each case, the Frobenius error decays as s increases, although larger matrices require larger sample sizes, as reflected by the upward shift of the curves as d is increased.

Next, we compare the performance of the max-norm based regularization with that of the trace-norm using the same criterion as in Davenport et al. (2012). More specifically, the target matrix M^* is constructed at random by generating $M = LR^T$, where L and R are $d \times r$ matrices with i.i.d. entries drawn from Uniform $[-1/2, 1/2]$, so that $\text{rank}(M^*) = r$. It is then scaled such that $\|M^*\|_\infty = 1$, while in the last case, M^* is formed such that $\|M^*\|_F/d = 1$. As before, we focus on the Gaussian conditional model but with noise level σ varies from 10^{-3} to 10, and set $d = 500$, $r = 1$ and $s = 0.15d^2$, which is exactly the same case studied in Davenport et al. (2012). We plot in Figure 2 the squared Frobenius norm of the error (normalized by the norm of the underlying matrix M^*) over a range of different values of noise level σ on a logarithmic scale. As evident in Figure 2, the max-norm based regularization performs slightly but consistently better than the trace-norm, except on the one point where $\sigma = \log_{10}(0.25)$. Also, we see that for both methods, the performance is poor when the noise is either too little or too much.

In the third experiment, we consider matrices with dimension $d = 200$ and choose a moderate level of noise, that is, $\sigma = \log_{10}(-0.75)$, according to previous experiences. Figure 3 plots the relative Frobenius norm of the error versus the sample size s for three different matrix ranks, $r \in \{3, 5, 10\}$. Indeed, larger rank means larger intrinsic dimension of the problem, and thus increases the difficulty of any reconstruction procedure.

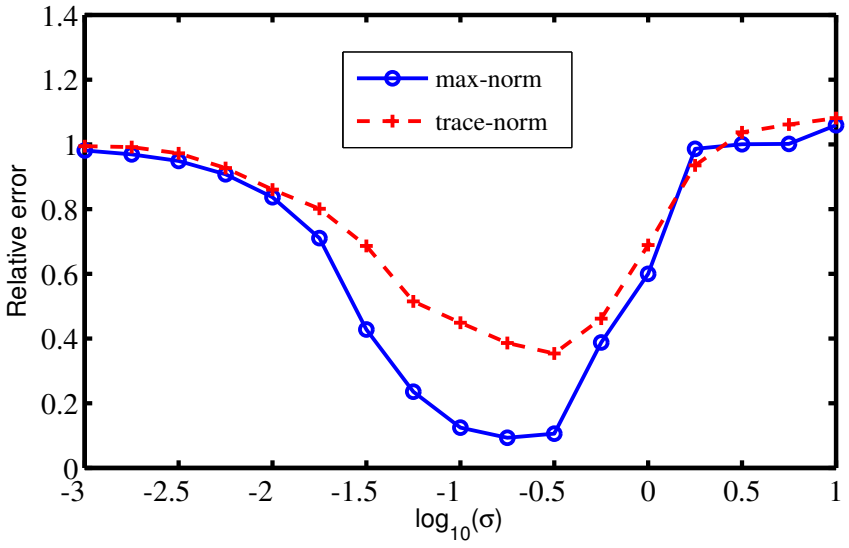


Figure 2: Plot of the relative Frobenius error $\|\widehat{M} - M^*\|_F^2 / \|M^*\|_F^2$ versus the noise level σ on a logarithmic scale, with rank $r = 1$, using both max-norm and trace-norm constrained methods.

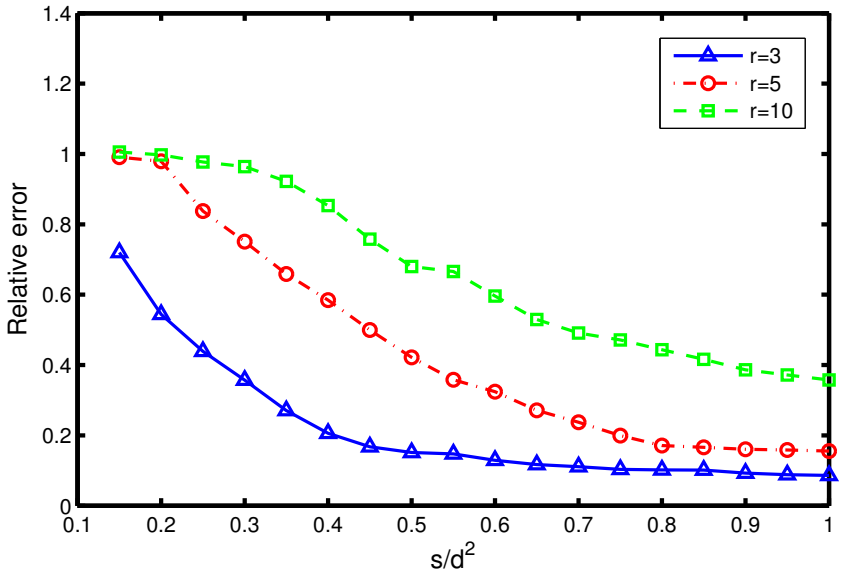


Figure 3: Plot of the relative Frobenius error versus the rescaled sample size s/d^2 for three different ranks $r \in \{3, 5, 10\}$, all with matrix size $d = 200$.

6. Discussion

This paper studies the problem of recovering a low-rank matrix based on highly quantized (to a single bit) noisy observation of a subset of entries. The problem was first formulated and analyzed by Davenport et al. (2012), where the authors consider approximately low-rank matrices in terms that the singular values belong to a scaled Schatten-1 ball. When the infinity norm of the unknown matrix M^* is bounded by a constant and its entries are observed uniformly in random, they show that M^* can be recovered from binary measurements accurately and efficiently.

Our theory, on the other hand, focuses on approximately low-rank matrices in the sense that unknown matrix belongs to certain max-norm ball. The unit max-norm ball is nearly the convex hull of rank-1 matrices whose entries are bounded in magnitude by 1, thus is a natural convex relaxation of low-rank matrices, particularly with bounded infinity norm. Allowing for non-uniform sampling, we show that the max-norm constrained maximum likelihood estimation is rate-optimal up to a constant factor, and that the corresponding convex program may be solved efficiently in polynomial time. An interesting question naturally arises that whether it is possible to push the theory further to cover exact low-rank matrix completion from noisy binary measurements.

The numerical study in Section 5 provides some evidence of the efficiency of the max-norm constraint approach in 1-bit matrix completion problem. More extensive experimental studies, applications to real data, and numerical comparisons with empirically weighted trace-norm method in a non-uniform scenario will be left as future work.

In our previous work (Cai and Zhou, 2013), we suggest to use max-norm constrained least square estimation to study standard matrix completion (from observations where additive noise is present) under a general sampling model. Similar error bounds are obtained, which are tight to within a constant. Comparing both results in the case of Gaussian noise demonstrates that as long as the signal-to-noise ratio remains constant, almost nothing is lost by quantizing to a single bit.

7. Proofs

We provide the proofs of the main results in this section.

7.1 Proof of Theorem 3

The proof of Theorem 3 is based on general excess risk bounds developed in Bartlett and Mendelson (2002) for empirical risk minimization when the loss function is Lipschitz. We regard matrix recovery as a prediction problem, that is, consider a matrix $M \in \mathbb{R}^{d_1 \times d_2}$ as a function: $[d_1] \times [d_2] \to \mathbb{R}$, that is, $M(k,l) = M_{k,l}$. Moreover, define a function $g(x;y)$ $\mathbb{R} \times \{\pm 1\} \mapsto \mathbb{R}$, which can be seen as a loss function:

$$g(x;y) = \mathbf{1}_{\{y=1\}} \log\left(\frac{1}{F(x)}\right) + \mathbf{1}_{\{y=-1\}} \log\left(\frac{1}{1-F(x)}\right).$$

For a subset $S = \{(i_1, j_1), ..., (i_n, j_n)\} \subseteq ([d_1] \times [d_2])^n$ of the observed entries of Y, let $\mathcal{D}_S(M;Y) = \frac{1}{n}\sum_{t=1}^n g(M_{i_t,j_t}; Y_{i_t,j_t}) = \frac{1}{n}\ell_S(M;Y)$ be the average empirical likelihood function, where the training set S is drawn i.i.d. according to Π (with replacement) on $[d_1] \times [d_2]$. Then we have

$$\mathcal{D}_\Pi(M;Y) := \mathbb{E}_{S\sim\Pi}[g(M_{i_t,j_t}; Y_{i_t,j_t})] = \sum_{(k,l)\in[d_1]\times[d_2]} \pi_{kl} \cdot g(M_{k,l}; Y_{k,l}).$$

Under condition (U3), we can consider g as a function: $[-\alpha,\alpha]\times\{\pm 1\}\to\mathbb{R}$, such that for any $y\in\{\pm 1\}$ fixed, $g(\cdot;y)$ is essentially an L_α-Lipschitz loss function. Also notice that in the current case, $Y_{i,j}$ take ± 1 values and appear only in indicator functions, $\mathbf{1}\{Y_{i,j}=1\}$ and $\mathbf{1}\{Y_{i,j}=-1\}$. Therefore, a combination of Theorem 8, (4) of Theorem 12 from Bartlett and Mendelson (2002) as well as the upper bound (6) on the Rademacher complexity of the unit max-norm ball yields that, for any $\delta>0$, the following inequality holds with probability at least $1-\delta$ over choosing a training set S of $2<n\le d_1d_2$ index pairs according to Π:

$$\begin{aligned}&\sup_{M\in K_{\max}(\alpha,R)}\big(\mathbb{E}_Y\mathcal{D}_\Pi(M;Y)-\mathbb{E}_Y\mathcal{D}_S(M;Y)\big)\\&\quad\le 17L_\alpha R\sqrt{\frac{d}{n}}+U_\alpha\sqrt{\frac{8\log(2/\delta)}{n}}:=R_n(\alpha,r;\delta).\end{aligned}\tag{31}$$

Since $\widehat{M}_{\max}$ is optimal and M^* is feasible to the optimization problem (13), we have

$$\mathcal{D}_S(\widehat{M}_{\max};Y)\le\mathcal{D}_S(M^*;Y)=\frac{1}{n}\sum_{t=1}^n g(M^*_{i_t,j_t};Y_{i_t,j_t}).$$

Because M^* has a fixed value which does not depend on S, the empirical likelihood term $\mathcal{D}_S(M^*;Y)$ is an unbiased estimator of $\mathcal{D}_\Pi(M^*;Y)$, that is,

$$\mathbb{E}_{S\sim\Pi}[\mathcal{D}_S(M^*;Y)]=\mathcal{D}_\Pi(M^*;Y).$$

Next, we will derive an upper bound on the deviation $\mathcal{D}_S(M^*;Y)-\mathcal{D}_\Pi(M^*;Y)$ that holds with high probability. To do this, let $A_1,\ldots,A_n$ be independent random variables taking values in $[d_1]\times[d_2]$ according to Π, that is, $\mathbb{P}[A_t=(k,l)]=\pi_{kl}$, $t=1,\ldots,n$, such that $\mathcal{D}_S(M^*;Y)=\frac{1}{n}\sum_{t=1}^n g(M_{A_t};Y_{A_t})$ and

$$\mathcal{D}_S(M^*;Y)-\mathcal{D}_\Pi(M^*;Y)=\frac{1}{n}\sum_{t=1}^n\big(g(M^*_{A_t};Y_{A_t})-\mathbb{E}[g(M^*_{A_t};Y_{A_t})]\big).$$

Then we apply the Hoeffding's inequality to the random variables $Z_{A_t}:=g(M^*_{A_t};Y_{A_t})-\mathbb{E}[g(M^*_{A_t};Y_{A_t})]$, conditionally on Y. Observe that $0\le g(M^*_{A_t};Y_{A_t})\le U_\alpha$ almost surely for all $1\le t\le n$, therefore for any $u>0$, we have

$$\mathbb{P}_{S\sim\Pi}\big\{\mathcal{D}_S(M^*;Y)-\mathcal{D}_\Pi(M^*;Y)>u\big\}\le\exp\Big(-\frac{2nu^2}{U_\alpha^2}\Big),\tag{32}$$

which in turn implies that that with probability at least $1-\delta$ over choosing a subset S according to Π,

$$\mathcal{D}_S(M^*;Y)-\mathcal{D}_\Pi(M^*;Y)\le U_\alpha\sqrt{\frac{\log(1/\delta)}{2n}}.\tag{33}$$

Putting pieces together, we get

$$\begin{aligned}&\mathbb{E}_Y\big[\mathcal{D}_\Pi(\widehat{M}_{\max};Y)-\mathcal{D}_\Pi(M^*;Y)\big]\\&\quad=\ \mathbb{E}_Y\big[\mathcal{D}_\Pi(\widehat{M}_{\max};Y)-\mathcal{D}_S(M^*;Y)\big]+\mathbb{E}_Y\big[\mathcal{D}_S(M^*;Y)-\mathcal{D}_\Pi(M^*;Y)\big]\\&\quad\le\ \mathbb{E}_Y\big[\mathcal{D}_\Pi(\widehat{M}_{\max};Y)-\mathcal{D}_S(\widehat{M}_{\max};Y)\big]+\mathbb{E}_Y\big[\mathcal{D}_S(M^*;Y)-\mathcal{D}_\Pi(M^*;Y)\big]\\&\quad\le\ \sup_{M\in K_{\max}(\alpha,R)}\big\{\mathbb{E}_Y[\mathcal{D}_\Pi(M;Y)]-\mathbb{E}_Y[\mathcal{D}_S(M;Y)]\big\}\\&\qquad+\mathbb{E}_Y\big[\mathcal{D}_S(M^*;Y)-\mathcal{D}_\Pi(M^*;Y)\big].\end{aligned}\tag{34}$$

Moreover, observe that the left-hand side of (34) is equal to

$$\mathbb{E}_Y\left[\mathcal{D}_\Pi(\widehat{M}_{\max};Y)-\mathcal{D}_\Pi(M^*;Y)\right] = \sum_{(k,l)\in[d_1]\times[d_2]}\pi_{kl}\left[F(M^*_{k,l})\log\left(\frac{F(M^*_{k,l})}{F((\widehat{M}_{\max})_{k,l})}\right)+(\bar{F}(M^*_{k,l}))\log\left(\frac{\bar{F}(M^*_{k,l})}{\bar{F}((\widehat{M}_{\max})_{k,l})}\right)\right],$$

which is the weighted Kullback-Leibler divergence between matrices $F(M)$ and $F(\widehat{M}_{\max})$, denoted by $\mathbb{K}_\Pi(F(M)\|F(\widehat{M}_{\max}))$, where

$$\bar{F}(\cdot):=1-F(\cdot) \quad \text{and} \quad F(M):=(F(M_{k,l}))_{d_1\times d_2}.$$

This, combined with (31), (33) and (34) imply that for any $\delta>0$, the following inequality holds with probability at least $1-\delta$ over S:

$$\mathbb{K}_\Pi(F(M^*)\|F(\widehat{M}_{\max}))\le R_n(\alpha,r;\delta/2)+U_\alpha\sqrt{\frac{\log(2/\delta)}{2n}}. \tag{35}$$

Together, (8), (35) and Lemma 8 below establish (16).

Lemma 8 (Lemma 2, Davenport et al., 2012) *Let F be an arbitrary differentiable function, and s,t are two real numbers satisfying $|s|,|t|\le\alpha$. Then*

$$d_H^2(F(s);F(t))\ge\inf_{|x|\le\alpha}\frac{(F'(x))^2}{8F(x)(1-F(x))}\cdot(s-t)^2$$

The proof of Theorem 3 is now completed. ■

7.2 Proof of Theorem 5

The proof for the lower bound follows an information-theoretic method based on Fano's inequality (Cover and Thomas, 1991), as used in the proof of Theorem 3 in Davenport et al. (2012). To begin with, we have the following lemma which guarantees the existence of a suitably large packing set for $K_{\max}(\alpha,R)$ in the Frobenius norm. The proof follows from Lemma 3 of Davenport et al. (2012) with a simple modification, see, for example, the proof of Lemma 3.1 in Cai and Zhou (2013).

Lemma 9 *Let $r=(R/\alpha)^2$ and $\gamma\le 1$ be such that $r\le\gamma^2\min(d_1,d_2)$ is an integer. There exists a subset $\mathcal{S}(\alpha,\gamma)\subseteq K_{\max}(\alpha,R)$ with cardinality*

$$|\mathcal{S}(\alpha,\gamma)|=\left[\exp\left(\frac{r\max(d_1,d_2)}{16\gamma^2}\right)\right]+1$$

and with the following properties:

(i) *For any $N\in\mathcal{S}(\alpha,\gamma)$,* $\operatorname{rank}(N)\le\frac{r}{\gamma^2}$ and $N_{k,l}\in\{\pm\gamma\alpha/2\}$, such that

$$\|N\|_\infty=\frac{\gamma\alpha}{2},\ \ \frac{1}{d_1d_2}\|N\|_F^2=\frac{\gamma^2\alpha^2}{4}.$$

(ii) For any two distinct $N^k, N^l \in \mathcal{S}(\alpha,\gamma)$,

$$\frac{1}{d_1 d_2}\|N^k - N^l\|_F^2 > \frac{\gamma^2\alpha^2}{8}.$$

Then we construct the packing set $\mathcal{M}$ by letting

$$\mathcal{M} = \Big\{N + \alpha(1-\gamma/2)E_{d_1,d_2} : N \in \mathcal{S}(\alpha,\gamma)\Big\}, \tag{36}$$

where $E_{d_1,d_2} \in \mathbb{R}^{d_1\times d_2}$ is such that the $(d_1,d_2)^{th}$ entry equals one and others are zero. Clearly, $|\mathcal{M}| = |\mathcal{S}(\alpha,\gamma)|$. Moreover, for any $M \in \mathcal{M}$, $M_{k,l} \in \{\alpha, (1-\gamma)\alpha\}$ by the construction of $\mathcal{S}(\alpha,\gamma)$ and (36), and

$$\|M\|_{\max} = \|N + \alpha(1-\gamma/2)E_{d_1,d_2}\|_{\max} \le \frac{\alpha\sqrt{r}}{2} + \alpha(1-\gamma/2) \le \alpha\sqrt{r},$$

provided that $r \ge 4$. Therefore, $\mathcal{M}$ is indeed a δ-packing of $K_{\max}(\alpha,R)$ in the Frobenius metric with

$$\delta^2 = \frac{\alpha^2\gamma^2 d_1 d_2}{8},$$

that is, for any two distinct $M, M' \in \mathcal{M}$, we have $\|M - M'\|_F \ge \delta$.

Next, a standard argument (Yang and Barron, 1999; Yu, 1997) yields a lower bound on the $\|\cdot\|_F$-risk in terms of the error in a multi-way hypothesis testing problem. More concretely,

$$\inf_{\widehat{M}} \max_{M\in K_{\max}(\alpha,R)} \mathbb{E}\|\widehat{M} - M\|_F^2 \ge \frac{\delta^2}{4}\min_{\widetilde{M}} \mathbb{P}(\widetilde{M} \ne M^\star),$$

where the random variable $M^\star \in \mathbb{R}^{d_1\times d_2}$ is uniformly distributed over the packing set $\mathcal{M}$, and the minimum is carried out over all estimators $\widetilde{M}$ taking values in $\mathcal{M}$. Applying Fano's inequality (Cover and Thomas, 1991) gives the lower bound

$$\mathbb{P}(\widetilde{M} \ne M^\star) \ge 1 - \frac{I(M^\star; Y_S) + \log 2}{\log|\mathcal{M}|}, \tag{37}$$

where $I(M^\star; Y_S)$ denotes the mutual information between the random parameter $M^\star$ in $\mathcal{M}$ and the observation matrix Y_S. Following the proof of Theorem 3 in Davenport et al. (2012), we could bound $I(M^\star; Y_S)$ as follows:

$$\begin{aligned} I(M^\star; Y_S) &\le \max_{M,M'\in\mathcal{M}, M\ne M'} \mathbb{K}(Y_S|M\|Y_S|M') \\ &= \max_{M,M'\in\mathcal{M}, M\ne M'} \sum_{(k,l)\in S} \mathbb{K}(Y_{k,l}|M_{k,l}\|Y_{k,l}|M'_{k,l}) \\ &\le \frac{n[F(\alpha) - F((1-\gamma)\alpha)]^2}{F((1-\gamma)\alpha)[1-F((1-\gamma)\alpha)]} \le \frac{n\alpha^2\gamma^2}{\beta_{(1-\gamma)\alpha}}, \end{aligned}$$

where the last inequality holds provided that $F'(x)$ is decreasing on $(0,\infty)$. Substituting this into the Fano's inequality (37) yields

$$\mathbb{P}(\widetilde{M} \ne M^\star) \ge 1 - \Big(\frac{n\alpha^2\gamma^2}{\beta_{(1-\gamma)\alpha}} + \log 2\Big)\Big/\Big(\frac{r(d_1\vee d_2)}{16\gamma^2}\Big)$$

Recall that $\gamma^* > 0$ solves the equation (22):

$$\gamma^* = \min\left\{\frac{1}{2}, \frac{R^{1/2}}{\alpha}\left(\frac{\beta_{(1-\gamma^*)\alpha}(d_1 \vee d_2)}{32n}\right)^{1/4}\right\}.$$

Requiring

$$\frac{64\log(2)(\gamma^*)^2}{d_1 \vee d_2} \le r \le (d_1 \wedge d_2)(\gamma^*)^2,$$

which is guaranteed by (23), to ensure that this probability is least $1/4$. Consequently, we have

$$\inf_{\widehat{M}} \max_{M \in K_{\max}(\alpha,R)} \mathbb{E}\|\widehat{M} - M\|_F^2 \ge \frac{\alpha^2(\gamma^*)^2 d_1 d_2}{128},$$

which in turn implies (24). ■

7.3 Proof of Theorem 7

The proof of Theorem 7 modifies the proof of Theorem 3, therefore we only summarize the key steps in the following. Let $\{A_1, ..., A_n\} = \{(i_1, j_1), ..., (i_n, j_n)\}$ be independent random variables taking values in $[d_1] \times [d_2]$ according to Π, and recall that

$$\ell_S(M;Y) = \sum_{t=1}^{s}\Big[\mathbf{1}_{\{Y_{A_t}=1\}}\log\Big(\frac{1}{F(M_{A_t})}\Big) + \mathbf{1}_{\{Y_{A_t}=-1\}}\log\Big(\frac{1}{1-F(M_{A_t})}\Big)\Big].$$

According to Srebro et al. (2005) and the proof of Theorem 3, it suffices to derive an upper bound on

$$\Delta := \mathbb{E}\Big[\sup_{M \in K_{\Pi,*}} \sum_{t=1}^{n} \frac{\varepsilon_t}{\sqrt{\pi_{i_t\cdot}\pi_{\cdot j_t}}}(M_w)_{A_t}\Big] = \mathbb{E}\Big[\sup_{M \in K_*(\alpha,r)} \sum_{t=1}^{n} \frac{\varepsilon_t}{\sqrt{\pi_{i_t\cdot}\pi_{\cdot j_t}}} M_{A_t}\Big],$$

where ε_t are i.i.d. Rademacher random variables. Then it follows from (5) that

$$\begin{aligned}
\Delta &\le \alpha\sqrt{r}\cdot\mathbb{E}\Big[\sup_{\|u\|_2=\|v\|_2=1} \sum_{t=1}^{n} \frac{\varepsilon_t}{\sqrt{\pi_{i_t\cdot}\pi_{\cdot j_t}}} u_{i_t} v_{j_t}\Big] \\
&= \alpha\sqrt{r}\cdot\mathbb{E}\Big[\sup_{\|u\|_2=\|v\|_2=1} \sum_{i,j}\Big(\sum_{t:(i_t,j_t)=(i,j)} \frac{\varepsilon_t}{\sqrt{\pi_{i_t\cdot}\pi_{\cdot j_t}}}\Big) u_i v_j\Big] \\
&= \alpha\sqrt{r}\cdot\mathbb{E}\Big[\Big\|\sum_{t=1}^{n} \varepsilon_t \frac{e_{i_t} e_{j_t}^T}{\sqrt{\pi_{i_t\cdot}\pi_{\cdot j_t}}}\Big\|\Big].
\end{aligned}$$

An upper bound on the above spectral norm has been derived in Foygel et al. (2011) using a recent result of Tropp (2012). Let $Q_t = \varepsilon_t \frac{e_{i_t} e_{j_t}^T}{\sqrt{\pi_{i_t\cdot}\pi_{\cdot j_t}}} \in \mathbb{R}^{d_1 \times d_2}$ be i.i.d. random matrices with zero-mean, then the problem reduces to estimate $\mathbb{E}\|\sum_{t=1}^{s} Q_t\|$. Following the the proof of Foygel et al. (2011), we see that, under condition (26)

$$\mathbb{E}\Big\|\sum_{t=1}^{n} Q_t\Big\| \le C\Big(\sigma_1\sqrt{\log(d)} + \sigma_2\log(d)\Big)$$

with

$$\begin{aligned}\sigma_1 &= n\cdot\max\left\{\max_k\sum_l\frac{\pi_{kl}}{\pi_{k\cdot}\pi_{\cdot l}},\max_l\sum_k\frac{\pi_{kl}}{\pi_{k\cdot}\pi_{\cdot l}}\right\}\le\mu n\max\{d_1,d_2\},\\ \sigma_2 &= \max_{k,l}\frac{1}{\sqrt{\pi_{k\cdot}\pi_{\cdot l}}}\le\mu\sqrt{d_1d_2}.\end{aligned}$$

Putting above estimates together, we conclude that

$$\Delta\le C\alpha\sqrt{r}\Big(\sqrt{\mu n\max\{d_1,d_2\}\log(d)}+\mu\sqrt{d_1d_2}\log(d)\Big),$$

which in turn yields that for any $\delta\in(0,1)$, inequality

$$\begin{aligned}&\mathbb{K}_\Pi(F(M^*)\|F(\widehat{M}_{w,tr}))\\ &\quad\le\quad C\left\{L_\alpha\alpha\sqrt{\frac{\mu r\max\{d_1,d_2\}\log(d)}{n}}+U_\alpha\sqrt{\frac{\log(4/\delta)}{n}}\right\}\end{aligned}$$

holds with probability at least $1-\delta$, provided that $n\ge\mu\min\{d_1,d_2\}\log(d)$. ■

7.4 An Extension to Sampling Without Replacement

In this paper, we have focused on sampling with replacement. We shall show here that in the uniform sampling setting, the results obtained in this paper continue to hold if the (binary) entries are sampled without replacement. Recall that in the proof of Theorem 3, we let $A_1,...,A_n$ be random variables taking values in $[d_1]\times[d_2]$, $S=\{A_1,...,A_n\}$ and assume the A_t's are distributed uniformly and independently, that is, $S\sim\Pi=\{\pi_{kl}\}$ with $\pi_{kl}\equiv\frac{1}{d_1d_2}$. The purpose now is to prove that the arguments remain valid when the A_t's are selected without replacement, denoted by $S\sim\Pi_0$. In this notation, we have

$$\mathcal{D}_S=\frac{1}{n}\sum_{(i,j)\in S}g(M_{i,j};Y_{i,j})\quad\text{and}\quad\mathcal{D}_{\Pi_0}=\mathbb{E}_{S\sim\Pi_0}[\mathcal{D}_S]=\frac{1}{d_1d_2}\sum_{(k,l)}g(M_{k,l};Y_{k,l}).$$

By Lemma 3 in Foygel and Srebro (2011) and (31), for any $\delta>0$,

$$\sup_{M\in K_{\max}(\alpha,R)}\big(\mathbb{E}_Y\mathcal{D}_{\Pi_0}(M;Y)-\mathbb{E}_Y\mathcal{D}_S(M;Y)\big)\le 17L_\alpha R\sqrt{\frac{d}{n}}+U_\alpha\sqrt{\frac{8(\log(4n)+\log(2/\delta))}{n}}$$

holds with probability at least $1-\delta$ over choosing a training set S of $2<n\le d_1d_2$ index pairs according to Π_0. Next, observe that the large deviation bound (32) for the sum of independent bounded random variables is a direct consequence of Hoeffding's inequality. To see how inequality (32) may be extended to the current case, we start with a more general problem. Let $\mathcal{C}$ be a finite set with cardinality N. For $1\le n\le N$, let $X_1,...,X_n$ be independent random variables taking values in $\mathcal{C}$ uniformly at random, such that $(X_1,...,X_n)$ is a $\mathcal{C}^n$-valued random vector modeling sampling with replacement from $\mathcal{C}$. On the other hand, let $(Y_1,...,Y_n)$ be a $\mathcal{C}^n$-valued random vector sampled uniformly without replacement. Assume that X_i is centered and bounded, and write $S_X=\sum_{i=1}^nX_i$, $S_Y=\sum_{i=1}^nY_i$. Then a large deviation bound holds for S_X by Hoeffding's inequality. In the proof, the tail probability is bounded from above in terms of the moment-generating function, say, $m_X(\lambda)=$

$\mathbb{E}\exp(\lambda S_X)$. According to the notion of negative association (Joag-Dev and Proschan, 1983), it is well-known that $m_Y(\lambda) = \mathbb{E}\exp(\lambda S_Y) \leq m_X(\lambda)$, which in turn gives a similar large deviation bound for S_Y. Therefore, inequalities (32) and (33) are still valid if Π is replaced by Π_0. Keep all other arguments the same, we get the desired result.

Acknowledgments

The authors would like to thank Yaniv Plan for helpful discussions and for pointing out the importance of allowing non-uniform sampling. We also thank the Associate Editor and two referees for thorough and useful comments which have helped to improve the presentation of the paper. The research of Tony Cai was supported in part by NSF FRG Grant DMS-0854973, NSF Grant DMS-1208982, and NIH Grant R01 CA127334. A part of this work was done when the second author was visiting the Wharton Statistics Department of the University of Pennsylvania. He wishes to thank the institution and particularly Tony Cai for their hospitality.

References

A. Ai, A. Lapanowski, Y. Plan and R. Vershynin. One-bit compressed sensing with non-Gaussian measurements. *Linear Algebra and Its Applications*, forthcoming, 2013.

P. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, **3**:463–482, 2002.

P. Biswas, T. C. Lian, T. C. Wang and Y. Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Transactions on Sensor Networks*, **2**(2):188–220, 2006.

P. Boufounos and R. G. Baraniuk. 1-bit compressive sensing. In *Proceedings of the 42nd Annual Conference on Information Sciences and Systems*, pages 16–21, Princeton, NJ, March 2008.

S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming, Series B*, **95**(2):329–357, February 2003.

T. T. Cai and W.-X. Zhou Matrix completion via max-norm constrained optimization. *arXiv preprint arXiv:1303.0341*, 2013.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley and Sons, New York, 1991.

M. A. Davenport, Y. Plan, E. van den Berg and M. Wooters. 1-bit matrix completion. *arXiv preprint arXiv:1209.3672*, 2012.

R. Foygel, R. Salakhuidinov, O. Shamir and N. Srebro. Learning with the weighted trace-norm under arbitrary sampling distributions. *Advances in Neural Information Processing Systems (NIPS)*, **24**, pages 2133–2141, 2011.

R. Foygel and N. Srebro. Concentration-based guarantees for low-rank matrix reconstruction. *JMLR: Workshop and Conference Proceedings*, **19**:315–339, 2011.

D. Goldberg, D. Nichols, B. M. Oki and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, **35**(12):61–70, December 1992.

P. E. Green and Y. Wind. *Multiattribute Decisions in Marketing: A Measurement Approach*. Dryden Press, Hinsdale, IL, 1973.

D. Gross and V. Nesme. Note on sampling without replacing from a finite collection of matrices. *arXiv preprint arXiv:1001.2738*, 2010.

L. Jacques, J. N. Laska, P. T. Boufounos and R. G. Baraniuk. Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *arXiv preprint arXiv:1104.3160*, 2011.

M. Jaggi. Revisiting Frank-Wolfe: projection-free sparse convex optimization. *JMLR: Workshop and Conference Proceedings*, **28**(1):427–435, 2013.

G. J. O. Jameson. *Summing and Nuclear Norms in Banach Space Theory*. London Mathematical Society Student Texts, 8. Cambridge University Press, Cambridge, 1987.

K. Joag-Dev and F. Proschan. Negative association of random variables with applications. *Annals of Statistics*, **11**(1):286–295, 1983.

O. Klopp. Noisy low-rank matrix completion with general sampling distribution. *Bernoulli*, forthcoming, 2012.

V. Koltchinskii, K. Lounici and A. B. Tsybakov. Nuclear norm penalization and optimal rates for noisy low rank matrix completion. *Annals of Statistics*, **39**(5):2302–2329, 2011.

J. N. Laska and R. G. Baraniuk. Regime change: bit-depth versus measurement-rate in compressive sensing. *IEEE Transactions on Signal Processing*, **60**(7):3496–3505, 2012.

R. Latala. Some estimates of norms of random matrices. *Proceedings of the American Mathematical Society*, **133**(5):1273–1282, 2005.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces. Isoperimetry and Processes*. Springer-Verlag, Berlin, 2011.

J. Lee, B. Recht, R. Salakhutdinov, N. Srebro and J. Tropp. Practical large-scale optimization for max-norm regularization. *Advances in Neural Information Processing Systems (NIPS)*, **23**, 2010.

N. Linial, S. Mendelson, G. Schechtman and A. Shraibman. Complexity measures of sign measures. *Combinatorica*, **27**(4):439–463, 2007.

S. Negahban and M. J. Wainwright. Restricted strong convexity and weighted matrix completion: optimal bounds with noise. *Journal of Machine Learning Research*, **13**:1665–1697, 2012.

Y. Plan and R. Vershynin. Robust 1-bit compressed sensing and sparse logistic regression: a convex programming approach. *IEEE Transactions on Information Theory*, **59**(1):482–494, January, 2013a.

Y. Plan and R. Vershynin. One-bit compressed sensing by linear programming. *Communications on Pure and Applied Mathematics*, **66**(8):1275–1297, August, 2013b.

R. Salakhutdinov and N. Srebro. Collaborative filtering in a non-uniform world: learning with the weighted trace norm. *Advances in Neural Information Processing Systems (NIPS)*, **23**, December, 2010.

Y. Seginer. The expected norm of random matrices. *Combinatorics, Probability and Computing*, **9**(2):149–166, March, 2000.

I. Spence and D. W. Domoney. Single subject incomplete designs for nonmetric multidimensional scaling. *Psychometrika*, **39**(4):469–490, December, 1974.

N. Srebro, J. Rennie and T. Jaakkola. Maximum-margin matrix factorization. *Advances in Neural Information Processing Systems (NIPS)*, **17**, pages 1329–1336. MIT Press, 2005.

N. Srebro and A. Shraibman. Rank, trace-norm and max-norm. In *Learning theory, Proceedings of COLT-2005. Lecture Notes in Computer Science*, **3559**:545–560, Springer, Berlin, 2005.

N. Srebro, K. Sridharan and A. Tewari. Optimistic rates for learning with a smooth loss. *arXiv preprint arXiv:1009.3896*, 2010.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, **12**(4):389–434, August, 2012.

Y. Yang and A. Barron. Information-theoretic determination of minimax rates of convergence. *Annals of Statistics*, **27**(5):1564–1599, 1999.

B. Yu. Assouad, Fano and Le Cam. *Research Papers in Probability and Statistics: Festschrift for Lucien Le Cam*, pages 423–435.

Journal of Machine Learning Research 14 (2013) 3649-3681 Submitted 4/12; Revised 7/13; Published 12/13

Efficient Program Synthesis Using Constraint Satisfaction in Inductive Logic Programming

John Ahlgren AHLGREN@EE.CITYU.EDU.HK
Shiu Yin Yuen KELVINY.EE@CITYU.EDU.HK
Department of Electronic Engineering
City University of Hong Kong
Hong Kong, China

Editor: Luc De Raedt

Abstract

We present NrSample, a framework for program synthesis in inductive logic programming. NrSample uses propositional logic constraints to exclude undesirable candidates from the search. This is achieved by representing constraints as propositional formulae and solving the associated constraint satisfaction problem. We present a variety of such constraints: pruning, input-output, functional (arithmetic), and variable splitting. NrSample is also capable of detecting search space exhaustion, leading to further speedups in clause induction and optimality. We benchmark NrSample against enumeration search (Aleph's default) and Progol's A^* search in the context of program synthesis. The results show that, on large program synthesis problems, NrSample induces between 1 and 1358 times faster than enumeration (236 times faster on average), always with similar or better accuracy. Compared to Progol A^*, NrSample is 18 times faster on average with similar or better accuracy except for two problems: one in which Progol A^* substantially sacrificed accuracy to induce faster, and one in which Progol A^* was a clear winner. Functional constraints provide a speedup of up to 53 times (21 times on average) with similar or better accuracy. We also benchmark using a few concept learning (non-program synthesis) problems. The results indicate that without strong constraints, the overhead of solving constraints is not compensated for.

Keywords: inductive logic programming, program synthesis, theory induction, constraint satisfaction, Boolean satisfiability problem

1. Introduction

Inductive logic programming (ILP) is a branch of machine learning that represents knowledge as first-order horn clauses. By using its expressive relational representation, it can overcome limitations inherent in propositional representations (Russell et al., 1996). ILP can be used for both concept learning and program synthesis, as the knowledge representation is at the same time declarative and procedural (Blackburn et al., 2006; Sterling and Shapiro, 1994).[1] In other words, induced solutions can both be regarded as human readable descriptions and as executable programs. Another advantage of ILP is its rigorous foundation in logic, making it suitable for theoretical analysis (Nienhuys-Cheng and de Wolf, 1997; Plotkin, 1970, 1971). A comprehensive survey of the ILP research field and its applications can be found in Muggleton et al. (2012).

1. In this paper, we use the term *concept learning* to refer to non-program synthesis problems, although program synthesis problems could also be considered concept learning problems.

State-of-the-art ILP systems such as Progol and Aleph use the technique of inverse entailment to induce theories (Muggleton, 1995; Srinivasan, 2001). Inverse entailment works by first constructing a bottom clause, from which all clauses that subsume it (or are supersets of it) become candidates. The search space hence becomes a lattice structure with a partial generality order, with the bodyless clause as top element and bottom clause as bottom element.

Mode declarations, as introduced in Muggleton (1995), may be used to constrain the search space further by specifying which variables are input and output, as well as requiring variables to be of a certain type. This input-output specification implicitly defines a logical constraint on the chaining of literals as the clause is computed from left to right. We describe mode declarations in Section 2.2.1. Mode declarations can be user provided or automatically constructed by analyzing the examples (McCreath and Sharma, 1995).

Inverse entailment is the method of constructing a lower bound on the search space. However, it does not force a certain ordering on the search. Thus various search strategies exist; some well known ones are Progol's A^* search (Muggleton, 1995), QG/GA (Muggleton and Tamaddoni-Nezhad, 2008), Aleph's enumeration (its default) (Srinivasan, 2001), and simulated annealing (Serrurier et al., 2004). What all these methods share in common is that the candidate generation mechanism employed is not adapted to avoid creation of candidates that violate the mode declarations. The mode declaration types are automatically handled by the bottom clause construction algorithm, but correct input-output chaining may be violated since candidates contain a subset of the bottom clause's literals. Such candidates are intended to be omitted by the syntactic bias a user provides using mode declarations, and should hence be considered redundant. One approach, taken by Aleph's enumeration, is to check input-output validity of the candidate before evaluation, but this still incurs the overhead of having to construct the candidate. Another approach, taken by algorithms using refinement operators, such as Progol's A^*, is to be indifferent about input-output violated candidates, evaluating all generated candidates. This is due to the fact that top-down (or bottom-up) searches may have to expand undesirable candidates in order to reach desirable ones. Both these approaches suffer from wasted computations as candidates in the search space violating the given mode declarations are still generated (and even evaluated).

In this paper, we present NrSample ("Non-redundant Sampling Algorithm"), a general constraint satisfaction framework for ILP that ensures only candidates that conform to mode declarations are generated. NrSample is implemented (along side other algorithms we use in our benchmarks) in our ILP system Atom.[2] This is to be contrasted with the aforementioned approaches, whereby a candidate is first generated and then tested for mode declaration correctness. We shall refer to candidates that are correct with respect to their mode declarations as *mode conformant*.

We achieve mode conformance by representing the input-output logic given by the mode declarations as propositional clauses (Russell et al., 1996). By solving the accompanying Boolean satisfiability problem (SAT), we obtain new candidates that by construction necessarily satisfy all constraints, and thus are mode conformant.

Moreover, other constraints naturally arise as a result of pruning the search space. Whenever a candidate is consistent (covers no negative examples), so are all specializations. Whenever a candidate is inconsistent (covers at least one negative example), so are all generalizations. As we will show, such pruning constraints are also easily represented as propositional clauses. More conventional approaches may use memorization to avoid evaluating candidates more than once, but

2. Source code available at `http://ahlgren.info/research/atom`, mirror available at `http://www.ee.cityu.edu.hk/~syyuen/Public/Code.html`.

the candidates may still be constructed multiple times. We shall refer to candidates that violate the constraints (mode declarations, pruning, or any other constraints) as *invalid*, and the rest as *valid*.

We represent our constraints as propositional clauses. Any SAT solver may then be used to solve the constraints and obtain a model representing a valid candidate. For efficiency, we use the Chaff algorithm (Moskewicz et al., 2001), which forms the basis of many state-of-the-art DPLL-based algorithms (Davis-Putnam-Logemann-Loveland algorithm) (Davis et al., 1962). Various selection strategies may be used within the SAT solver to guide the search into interesting regions early. By using a *complete* SAT solver (all DPLL-based algorithms are complete), any search strategy is also guaranteed to terminate as soon as all non-redundant candidates have been explored.

A survey and discussion of program synthesis in ILP can be found in Flener and Yılmaz (1999). Constraints have been used in ILP before, although not to constrain the bottom clause literals. Constraint logic programming with ILP (Sebag and Rouveirol, 1996) turns negative examples into constraints, primarily as a means of dealing with numerical constraints. Theta-subsumption with constraint satisfaction (Maloberti and Sebag, 2004) uses constraints to speed up theta-subsumption testing (during coverage). Condensed representations (De Raedt and Ramon, 2004) are used to deal with redundancies in frequent Datalog queries. In the context of theory induction, schemata may be used to guide the construction of logic programs (Flener et al., 2000). Our framework differs from all the above in that the constraints specify which of a bottom clause's literals must be (or may not be) present in any candidate. The constraints are thus propositional in nature, and relative to a computed bottom clause. Our approach attempts to minimize the amount of redundant candidates *constructed*.

This paper is organized as follows. First, we describe propositional constraints and candidate generation in Section 2. Next, Section 3 describes our search algorithm that uses these constraints. In Section 4, we perform benchmarks to test our hypothesis that NrSample can outperform generate-and-test methods, in particular when the search space is large. Finally, Section 5 concludes the paper.

2. Propositional Constraints and SAT Solving

Although our framework enables the use of arbitrary propositional constraints to define redundancies during an ILP search for candidates, we will present two common instances of such constraints.

Firstly, NrSample constrains candidate generation to only those that conform to the mode declarations. Secondly, it prunes too general or specific solutions after each evaluation: which depends on whether the candidate is consistent or inconsistent.

NrSample achieves non-redundancy by storing all the constraints as propositional formulae. Candidate solutions during a search contain a subset of the bottom clause's literals (for the subsumption order, we discuss variable splitting in Section 2.4). Hence, each candidate can be represented as a bit string where bit b_i signifies occurrence of body literal at position i or lack thereof. Seen from a slightly different perspective, we represent the occurrence or non-occurrence of a literal at position i in the bottom clause by a propositional variable b_i and $\neg b_i$, respectively. With respect to a bottom clause, there is thus a one-to-one correspondence between each Boolean assignment and candidate solution. The propositional constraints correspond to the syntactic bias induced by the user generated mode declarations and the search lattice pruning. The idea is that a solution is evaluated for coverage if and only if it corresponds to a propositional model for the constraints (a variable assignment that makes all constraints true). This enables us to invoke a SAT solver to

retrieve models for the constraints, which are then easily converted into candidate solutions. After each candidate is evaluated, pruning constraints related to generality order redundancies are added.

In the following sections, we describe how we create new constraints and retrieve models from the constraint database.

2.1 Clauses and Propositional Formulae

We start by defining the notion of a search space *candidate*.

Definition 1 *Let B be a definite clause (a clause with a head). A clause C is a* candidate from B *if and only if C's head is the same as B's head, and C's body is a subsequence of B's body.*

Here, B is intended to be the bottom clause, and C a clause that is created by possibly removing body literals of B. Note that B itself is a candidate from B. Usually, we are only interested in candidates from a specific bottom clause, in which case we omit the reference to B. Note also that the definition of subsequence forces the body literals of C to appear in the same order as those of B.

Example 1 *With bottom clause $B = h(X,Y) \leftarrow q_1(X,Z), q_2(Z,Y), q_3(X,Y,Z)$, the clause $C = h(X,Y) \leftarrow q_1(X,Z), q_3(X,Y,Z)$ is a candidate from B since they have the same head and $(q_1(X,Z), q_3(X,Y,Z))$ is a subsequence of $(q_1(X,Z), q_2(Z,Y), q_3(X,Y,Z))$. $D = h(X,Y) \leftarrow q_3(X,Y,Z), q_1(X,Z)$ is however not a candidate from B, since the sequence $(q_3(X,Y,Z), q_1(X,Z))$ is not a subsequence of $(q_1(X,Y), q_2(Y,Z), q_3(X,Y,Z))$.*

Definition 1 defines a search space lattice of candidates spanned by subset order.[3] The more general subsumption order can be explored using variable splitting, as discussed in Section 2.4.

To create a one-to-one correspondence between first-order horn clauses and propositional formulae with respect to a bottom clause, we represent each literal of the bottom clause from left to right as propositional variables $b_1, b_2, \ldots, b_n$, where n is the number of literals in the body of the bottom clause. Given a clause C which has some of the body literals in B, the propositional formula for C is then the conjunction of all propositional variables in B with positive sign if they occur in C and negative sign otherwise.

Definition 2 *Let C be a candidate from B, where B has n body literals. The propositional formula for C is $P_C^B = \bigwedge_{i=1}^{n} l_i$, where $l_i = b_i$ if C contains the ith literal in B, and $l_i = \neg b_i$ otherwise. We write P_C when there is no confusion about which bottom clause we are referring to.*

Note that P_C is a conjunction of *all* literals $b_1, \ldots, b_n$, where n is the number of body literals in B. In particular, non-occurrence of a literal has to be specified with its corresponding negative propositional literal. This way, each propositional formula has precisely one model, corresponding to the candidate itself.

Example 2 *Continuing from our previous example, $P_C^B = b_1 \wedge \neg b_2 \wedge b_3$ and $P_B^B = b_1 \wedge b_2 \wedge b_3$.*

The purpose of representing clauses as propositional formulae (with respect to some bottom clause) is that we can solve the formulae to acquire a model, which can then trivially be converted into a candidate.

3. Technically, by subsequence order, but as we do not consider re-orderings, there is no risk of confusion.

Definition 3 *Let B be a clause with n body literals, F^B a propositional formula containing a subset of the propositional variables $\{b_1,\ldots,b_n\}$, and M a model for F^B. The propositional formula (from B) generated by M is:*

$$P_M^B = \bigwedge_{i=1}^{n} l_i, \text{ where } l_i = \begin{cases} b_i & \text{if } M(b_i) = true \\ \neg b_i & \text{if } M(b_i) = false \end{cases}.$$

If M generates the propositional formula P_C, C is the candidate (from B) generated by M.

Example 3 *Let B be defined as in Example 1 and $F^B = b_1 \wedge (b_2 \vee b_3)$. Then $M = \{b_1 = true, b_2 = false, b_3 = true\}$ is a model for F^B. The propositional formula generated by M is $b_1 \wedge \neg b_2 \wedge b_3$. The candidate generated by M is $h(X,Y) \leftarrow q_1(X,Z), q_3(X,Y,Z)$.*

Usually, we are not only interested in what candidate a specific model generates, but rather, all candidates generated by all models of a propositional formula.

Definition 4 *Let B be a clause with n body literals and F^B a propositional formula. The propositional formulae generated by F^B are the propositional formulae generated by all models of F^B. The candidates (from B) generated by F^B are the candidates corresponding to those propositional formulae.*

Example 4 *Let B and F^B be as in the previous example. In all models of F^B, b_1 is true, and at least one of b_2, b_3 is true. This gives 3 models for F^B, and the propositional formulae generated by F^B are $b_1 \wedge b_2 \wedge \neg b_3$, $b_1 \wedge \neg b_2 \wedge b_3$, and $b_1 \wedge b_2 \wedge b_3$. The candidates generated by F^B are $h(X,Y) \leftarrow q_1(X,Z), q_2(Z,Y)$, $h(X,Y) \leftarrow q_1(X,Z), q_3(X,Y,Z)$, and $h(X,Y) \leftarrow q_1(X,Z), q_2(Z,Y), q_3(X,Y,Z)$, respectively. Intuitively, the formula F^B tells us that a candidate must have the first literal ($q_1(X,Z)$) and at least one of the two that follow it. By looking at all models for F^B, we can retrieve these candidates explicitly.*

Our constraints are represented as a propositional formula (more specifically, a conjunction of clauses, as we will see later). The constraints represent which candidates are allowed, so we start with the constraint *true* to allow any candidate from *B*. To retrieve an allowed (non-redundant) candidate, we compute a model for the constraints using our SAT solver. This model then generates our candidate solution as of Definition 3. The set of all allowed candidates are those generated by our constraints as of Definition 4.

Example 5 *As an example of how constraints work, assume we want an algorithm that never generates a previously evaluated candidate. For each candidate C, P_C is the corresponding propositional formula. Since our constraints specify the set of allowed candidates through its models, adding $\neg P_C$ will prevent candidate C (and only C) from being generated again.*

Now we show which constraints to add in order to prevent mode violations and redundancies that occur due to the generality order. The latter includes blocking out already visited candidates as in the example above, but prunes more of the search space.

2.2 Mode Declaration Constraints

Intuitively, mode declarations show where the inputs and outputs of a literal are located, and which types it takes. The bottom clause is then constructed using the mode declarations so that no literal is introduced until all its inputs have been instantiated by previous literals. Here we briefly explain mode declarations, referring the reader to Muggleton (1995) for a more general description.

2.2.1 MODE DECLARATIONS EXPLAINED

A mode declaration is a specification that limits the usage of variables (in the literals of clauses) to some specific type (for example, numbers or lists) and specify whether they will be used as input or output. Input variables need to be instantiated (computed) by a previous literal in the sequence of body literals, or by the head. Output variables occurring in body literals do not have any restrictions: they may or may not have been previously instantiated. Output variables in the head need to be instantiated by the body, or by an input variable to the head.

Mode declarations are used to restrict the number of literals in the bottom clause—by requiring input-instantiation—as well as providing computational speedup, by restricting attention to only the relevant data types. These aspects are being taken care of by the bottom clause construction algorithm. We introduce the syntax and semantics of mode declarations using an example of program synthesis.

Example 6 *Consider the mode declaration* $modeh(*, member(-constant, +list))$. *modeh refers to the head of a clause, as opposed to an atom used in the body. The first argument to modeh—where the asterisk is located—is where the maximum number of query answers is specified. This is known as the* recall. *The asterisk specifies that there is no upper limit (infinitely many), which is clearly the case for list membership, as there is no bound on the number of elements it may contain. When querying for the parents of a person, clearly the recall can be set to two (and for grandparents, to four). Recall does not interfere with our constraint framework: they can be used or ignored, and we will therefore make no further mention of them in this paper. The mode declaration declares a predicate member/2 (the* 2 *specifies the arity), and when it is used in the head of a clause, it outputs a constant (indicated by the minus sign) and requires a list as input (indicated by the plus sign). Thus we expect* $member(E, [3,5])$ *to be a valid query, since the list* $[3,5]$ *is used as input. On the other hand, the query* $member(3, L)$ *is invalid, since L is not instantiated.*

Now consider adding the mode declaration $modeb(*, +list = [-constant|-list])$. *This declares =/2 as a body literal (modeb) with a list on the left hand side as input, to obtain the head and tail of the list as outputs on the right hand side. Intuitively, this mode declaration introduces a predicate useful for splitting a list into its head and tail.*

In conjunction with the modeh declaration above, we may now generate the mode conformant clause $member(E, L) \leftarrow L = [E|T]$, *which is the proper base case definition of list membership. The occurrence of L in the head is as input (given by the modeh declaration), so L is already instantiated in the body literal as required. Also, E in the head is required to be output, a requirement fulfilled by the body literal as it instantiates E.*

An example of a clause that is not mode conformant is $member(E, L) \leftarrow X = [E|T]$, *since X is not an input type (it has not previously been instantiated). Another example of a non-conformant clause is* $member(E, L) \leftarrow L = [H|T]$, *this time because E is declared to be an output variable, but never instantiated (we obtain no meaningful answer to a query such as* $member(E, [2])$, *since E is never computed in the clause).*

Mode declarations can also be used to restrict the search itself, since candidates, by containing subsequences of the bottom clause's literals, may break an input-output chain. The next example illustrates this.

Example 7 *Assume we have computed bottom clause*

$$member(A,B) \leftarrow B = [C|D], member(A,D), member(C,B)$$

from an example. The mode declarations are

$$\begin{array}{l} modeh(*, member(+const, +clist)), \\ modeb(*, +clist = [-const|-clist]), and \\ modeb(*, member(+const, +clist)). \end{array}$$

In this case, A and B are both already instantiated in the head, as specified by the mode declarations (the plus signs).

The clause $member(A,B) \leftarrow member(A,D)$ *is a candidate, albeit not a mode conformant one, since D is never instantiated before appearing in the body literal (as required by the third mode declaration). We would need to include the bottom clause literal* $B = [C|D]$ *first, as it is the only literal that would instantiate D. Including both, we would then have the proper recursive clause for list membership:*

$$member(A,B) \leftarrow B = [C|D], member(A,D).$$

In this case we get a simple rule of the form "if the bottom clause's second body literal is included in a candidate, so must the first". In general we may have more than one possible instantiation, and literals may contain more than one input variable, making the rules more complex. In the next section, we work through the logic of mode declarations in detail.

2.2.2 MODE DECLARATIONS AS PROPOSITIONAL CLAUSES

A candidate is obtained from the bottom clause by taking a subsequence of the bottom clause's body literals and copying the head. As we have seen in Example 7, there is no guarantee that a randomly chosen candidate will respect the mode declarations. A common solution to this is to check for validity before evaluating a candidate. Our goal is to avoid the generation of unnecessary candidates in the first place.

For our purposes, there are two aspects of input-output chaining that affect the constraints:

1. Each body literal input must be instantiated from previous body literal outputs or from inputs to the head (inputs to the head are instantiated by the query).

2. The head outputs must be instantiated from head inputs or body literal outputs.

Definition 5 *Let C be a clause with n body literals. Let* I_i *and* O_i *be the set of input and output variables of the ith literal, respectively (as defined by the mode declarations). Denote the input and output variables of h by* I_h *and* O_h*, respectively. C is* input-instantiated *if and only if for all* $v \in I_i$*, we have* $v \in I_h$ *or* $v \in O_k$ *for some* $k < i$*. C is* output-instantiated *if and only if, for all* $v \in O_h$*, we have* $v \in I_h$ *or* $v \in O_k$ *for some k. C is* mode conformant *if and only if C is both input- and output-instantiated.*

Example 8 *With mode declarations* $modeh(*, h(+any, -any))$, $modeb(*, q_1(+any, -any))$ *and* $modeb(*, q_2(+any, -any))$, *the clause* $C = h(X,Z) \leftarrow q_1(X,Y), q_2(Y,X)$ *is input-instantiated, since* X *in* $q_1(X,Y)$ *grabs its input from the head input, and* Y *in* $q_2(Y,X)$ *grabs its input from the output of* $q_1(X,Y)$ *(which appears before* $q_2(Y,X)$*). It is however not output-instantiated, since the head output* Z *does not grab output from any output of the body literals (and is not in the head as input). Essentially, this means that in a query such as* $h(5,Ans)$, *the* 5 *will be "propagated" through all literals (*C *is input-instantiated), but our query variable Ans will not be bound (*C *is not output-instantiated). The clause* $D = h(X,Z) \leftarrow q_1(Y,Z)$ *is output-instantiated since* Z *in the head grabs output from* $q_1(Y,Z)$, *but not input-instantiated since* Y *in* $q_1(Y,Z)$ *is not instantiated. The clause* $E = h(X,Z) \leftarrow q_1(X,Y), q_2(Y,Z)$ *is both input- and output-instantiated, and hence mode conformant. Finally,* $F = h(X,Z) \leftarrow q_1(A,B)$ *is neither input- nor output-instantiated.*

Lemma 6 Given mode declarations, a bottom clause is always input-instantiated but not necessarily output-instantiated.

Proof Input-instantiation is a direct consequence of the way in which the bottom clause is constructed: we only add body literals when all their inputs have been instantiated by the head or previous body literals. To see that a bottom clause may not be output-instantiated, it is enough to consider a mode head declaration in which the output does not correspond to any body literal outputs: $modeh(*, h(+type1, -type2))$, $modeb(*, b(+type1, -type1))$. With type definitions $type1(a) \wedge type2(b)$, example $h(a,b)$ and background knowledge $b(a,a)$, we get the bottom clause $B = h(X,Y) \leftarrow b(X,X)$. B is indeed input-instantiated (X is instantiated in the head), but not output-instantiated since Y has no instantiation. ■

Our implementation of mode constraints is straightforward. We simply mimic the logic behind Definition 5. Informally, for each input variable v of a literal $q_i(\ldots)$, we require that it appears in at least one previous literal (as output) or the head (as input). Since the bottom clause head is always present in a candidate, no constraints apply when an input can be caught from the head. For each output variable v of the head that is not also an input to the head, we require that it appears in at least one output variable in a body literal.

Definition 7 *Let* B *be a bottom clause with* n *body literals. Let* I_i *and* O_i *be the input and output variables of body literal* i *(as given by the mode declarations), respectively. Similarly, denote by* I_h *and* O_h *the input and output variables of the head (as given by the modeh declaration), respectively. The mode constraints* F *of bottom clause* B *is a conjunction of clauses, constructed as follows:*

1. *For each* $v \in I_i$, $v \notin I_h$, *include clause* $\{\neg b_i\} \cup \{b_j : j < i, v \in O_j\}$.
2. *For each* $v \in O_h$, $v \notin I_h$, *include clause* $\{b_j : v \in O_j, j \in \{1,\ldots,n\}\}$.
3. *No other clauses are included.*

Note that if an output variable of B's head cannot be instantiated by any body literals, F will contain the empty clause (generated by the second rule) and therefore be inconsistent. This is correct, since no candidate from B will be output-instantiated.

The following theorem, which we prove in Appendix A, shows that our mode constraints are sound and complete.

Theorem 8 *Let F be a propositional formula for the mode constraints of B. Let C be a candidate from B. F generates C if and only if C is mode conformant.*

Proof See Appendix A. ■

2.3 Pruning Constraints

A candidate solution C is typically evaluated by comparing it against the set of all positive and negative examples. If the candidate covers a negative example, it is said to be *inconsistent*, otherwise *consistent*. Since our search lattice defines a subset order on the literals, where the top element is the empty clause and the bottom element is the bottom clause, the generality order is the subset order for body literals.

Definition 9 *Let C and D be candidates from B. We say that C is* more general *than D (with respect to B) if and only if C's body is a subsequence of D's body. We write $C \subseteq_B D$, omitting B whenever the bottom clause is clear from context. If C is more general than D, D is* more specific *than C.*

The following is a trivial consequence of the subset relation.

Lemma 10 *Let G and S be candidates. If $G \subseteq S$, then $G \Longrightarrow S$. The converse does not necessarily hold.*

Proof The subset relation is a special case of subsumption, for which the implication is well known (Nienhuys-Cheng and de Wolf, 1997). An application of self-resolution to any recursive clause demonstrates that the converse does not hold. ■

It follows that if $G \subseteq S$, G covers at least as many examples as S. In particular, if S covers a negative example and is thus inconsistent, all generalizations also cover that negative example, and are therefore also inconsistent. Since inconsistent solutions are of no interest to us, all generalizations of an inconsistent clause are redundant. On the other hand, if G covers no negative examples (G is consistent) and p positive examples, no specialization S can cover more than p positive examples. Hence the specializations of a consistent clause are redundant.

We would like to store information about which regions have been pruned during our search, so there will never be any redundant evaluation with respect to the subset order. To achieve this, we need to know what the pruning formulae look like. We start with an example.

Example 9 *Let us say that $B = h(X) \leftarrow q_1(X), q_2(X), q_3(X), q_4(X)$ and $C = h(X) \leftarrow q_3(X), q_4(X)$. If C is inconsistent, all generalizations will also be inconsistent. We can represent C by the bit string* 0011*, where a* 0 *or* 1 *at position i indicates absence or presence of literal i, respectively. All generalizations of C are given by clauses that do not contain $q_1(X)$ and $q_2(X)$, so we can represent them using the schema* 00$**$*. Logically, this corresponds to $\neg b_1 \wedge \neg b_2$. Conversely, all specializations of C are given by clauses that contain $q_3(X)$ and $q_4(X)$, so their schema is $**11$. Logically, this is the propositional formula $b_3 \wedge b_4$. This suggests the conjunction of all literals in C give all specializations, whereas the negated conjunction of all literals missing in C gives all generalizations.*

Next, we define such formulae.

Definition 11 *Let C be a candidate from B. For any candidate X from B, let V_X be all the propositional variables corresponding to the first-order body literals of B that occur in X. The* propositional formula for all generalizations of *C is*

$$P^B_{\uparrow C} = \bigwedge_{b_i \in V_B - V_C} \neg b_i.$$

The propositional formula for all specializations of *C is*

$$P^B_{\downarrow C} = \bigwedge_{b_i \in V_C} b_i.$$

We may then prove that our informal derivations are sound and complete.

Theorem 12 *Let C be a candidate from B.*

1. *$\neg P_{\uparrow C}$ generates G if and only if G is not a generalization of C.*
2. *$\neg P_{\downarrow C}$ generates S if and only if S is not a specialization of C.*

Proof See Appendix B. ■

The theorem is used in the following way: After evaluating a candidate C for coverage, we know whether it is consistent or inconsistent. If C is inconsistent, we prune all generalizations $P_{\uparrow C}$. This is done by inserting the requirement that no candidate generated by $P_{\uparrow C}$ is allowed: $\neg P_{\uparrow C} = \bigvee_{b_j \in V_B - V_C} b_j$. Similarly, if C is consistent, we add $\neg P_{\downarrow C} = \bigvee_{b_i \in V_C} \neg b_i$. Note that the constraints are always propositional clauses.

Example 10 *If $B = h(X) \leftarrow q_1(X), q_2(X), q_3(X), q_4(X)$ and $C = h(X) \leftarrow q_2(X), q_3(X)$, all generalizations of C are given by $\neg b_1 \wedge \neg b_4$ and all specializations by $b_2 \wedge b_3$. If C turns out to be inconsistent, all generalizations are also inconsistent, so we add the constraint $b_1 \vee b_4$. In particular, since any model either sets b_1 or b_4 to $true$, both C and the empty clause (top element) are excluded. If C turns out to be consistent, all specializations cover no more positive examples, so we add the constraint $\neg b_2 \vee \neg b_3$. Any model now sets b_2 or b_3 to $false$, which, for example, blocks both C and the bottom clause.*

2.4 Variable Splitting Constraints

Atom's and Progol's bottom clause construction assumes that equivalent ground terms are related when lifting instances. For example, when lifting

$$h(x,z) \leftarrow b_1(x,y), b_2(y,z)$$

the two occurrences of y are assumed to be more than a coincidence:

$$h(X,Z) \leftarrow b_1(X,Y), b_2(Y,Z).$$

Most of the time, this is not a problem since there will eventually be an example that reveals the coincidentally linked terms. However, Tamaddoni-Nezhad and Muggleton (2009) provide an example of a half adder, which cannot be induced with this restriction. Progol's A* solves this by

variable splitting during its search, but this method does not work when we represent clauses as binary strings, or, as in our case, propositional formulae. The solution, also taken by the Aleph ILP system (Srinivasan, 2001), is then to let the bottom clause represent these equality assumptions (variable splitting may also add many more literals to the bottom clause, but this will not affect our constraints).

As an optimization, we can ensure that no redundant candidates are generated due to these equalities, so we add constraints requiring that a variable splitting equality is used in at least one other literal. Since an added equality $X = Y$ matters only if both X and Y occur in the formula, we get the following variable splitting constraints.

Definition 13 *Let C be a candidate from B with a (possibly empty) prefix subsequence of k variable splitting equalities $V_i = W_i, i = 1,\ldots,k$ (V_i is a different variable from W_i). Let B_{V_i}, B_{W_i} be the set of variables corresponding to literals in C that are not variable splitting equalities ($i > k$) and contain V_i and W_i, respectively. For each b_i, $i = 1,\ldots,k$, we add two variable splitting constraints:*

1. $\{\neg b_i\} \cup B_{V_i}$, *and*
2. $\{\neg b_i\} \cup B_{W_i}$.

Note that input- and output-instantiation constraints will be added to the constraint database in the form of mode constraints; the equality constraints need only specify when equalities are redundant.

Example 11 *Let B be the bottom clause*

$$h(X,Y) \leftarrow V = W, q_2(X,V), q_3(Y,W), q_4(X,Y).$$

If C is a candidate from B containing the literal $V = W$, we require that both the variables V and W occur in C (other than as equalities generated from bottom clause construction). V occurs in the second literal and W in the third, so our propositional constraints are $\{\neg b_1, b_2\}$ and $\{\neg b_1, b_3\}$.

For instance, this prevents the candidate $h(X,Z) \leftarrow X = Y, b_2(X,A), b_3(A,Z)$ from being generated, as the equality is redundant since Y is never used. Instead, only the logically equivalent candidate obtained by removing the equality will be generated: $h(X,Z) \leftarrow b_2(X,A), b_3(A,Z)$.

2.5 Functional Constraints

Many predicates have the property that their truth value does not matter, only what they compute as output. We call such predicates *functional*. In particular, when chaining arithmetic operations as predicates, we are not concerned with their truth values, but only about obtaining a numeric output from inputs. (The predicate must also be *pure*, that is, free of any side effects, at least as far as can be measured by other predicates in use.)

The idea is that for functional predicates, we require that *at least one of its output variables occur in another literal*.

Example 12 *The is/2 operator computes an arithmetic expression given by its second argument and unifies the result with its first argument. With mode declaration*

$$modeb(1, is(-Real, +Expr), [functional])$$

we declare it functional. Then,

$$average(X,Y,A) \leftarrow S\,is\,X+Y, A\,is\,S/2$$

is a viable candidate to evaluate since the output S is used to compute the average of X and Y, and A is used as the final answer that is instantiated by the head. However,

$$average(X,Y,A) \leftarrow S\,is\,X+Y, D\,is\,Y-X, A\,is\,S/2$$

is logically equivalent but should not be a viable candidate, since the output D is never used (it occurs only as a singleton). Since the is-operator is always true when the left hand side variable is uninstantiated, 'D is Y − X' is a useless operation and the redundant candidate may safely be blocked from consideration.

Definition 14 *Let B be a bottom clause with n literals. A modeb declaration for a predicate can declare it a* functional predicate*. If the ith literal of B is declared a functional predicate, and none of its outputs occur in the head (as input or output), we define its corresponding* functional constraint *to be the propositional clause:*

$$\{\neg b_i\} \cup \{b_x : x \in \{1,2,\ldots,n\} \wedge x \neq i \wedge O_i \cap (I_x \cup O_x) \neq \emptyset\}.$$

If one or more of the ith literal's outputs occur in the head, no constraint is generated (clearly the predicate is always useful then).

Essentially, functional predicates filter out clauses which *only* have an effect through their predicate's truth value. Note that functional predicates are not required to have an effect on literals appearing later: its outputs may also be compared with previous known values, whether it may be inputs or outputs. Note that a functional predicate always generates at most one propositional clause, since we only require that *at least one* of its output variables occurs in another literal.

Example 13 *Elaborating more on Example 12, consider the bottom clause*

$$B = average(X,Y,A) \leftarrow S\,is\,X+Y, D\,is\,Y-X, A\,is\,D+X, A\,is\,S/2.$$

If we declare the predicate is/2 as functional, one constraint we would obtain is $\{\neg b_2, b_3\}$*. Intuitively, the constraint reflects the fact that if the second literal is used (D is Y − X), then D must appear elsewhere since otherwise the literal does nothing useful in functional terms. The only other occurrence of D is in the third literal, so* $b_2 \longrightarrow b_3$*. With this constraint, our candidate C from the previous example would never be generated, since the model for C sets* b_2 *to true and* b_3 *to false.*

Another constraint we would get is $\{\neg b_1, b_4\}$*. This constraint is due to the output S in the first literal, which must be used in its only other occurrence, the fourth literal. This one does however not prevent C, as* b_4 *is true in C's model.*

We do not get any constraints from the third and fourth literal output A; this is because A already occurs in the head.

3. NrSample: Constraint-Based Induction

NrSample's search algorithm is simple: after a bottom clause has been constructed, the mode constraints are generated. The SAT solver is then invoked to acquire a valid candidate, which is then evaluated. Pruning constraints are then added based on the candidate's consistency/inconsistency. The procedure of invoking the SAT solver to obtain candidates proceeds until a termination criterion is triggered (usually the maximum number of nodes to explore) or the search space is exhausted with respect to the constraints (no model can be acquired).

The induction procedure follows a sequential covering algorithm: The first available example is used to construct a bottom clause, leading to a space of possible candidates. If a viable candidate can be found from this search space, all positive examples it covers are removed. If no candidate is found, the example itself is moved to the generalized theory. This process repeats until all positive examples are covered. The bottom clause construction algorithm is equivalent to Progol's and is somewhat involved; we refer the reader to Muggleton (1995) for a detailed description. Briefly, it uses the mode declarations to compute the set of ground truths and lift them to variable instances. Each ground truth then becomes a body literal in the bottom clause. Pseudo-code for NrSample is given in Algorithm 1.

```
 1. While there is a positive example available:
 2.   Generate bottom clause from the example.
 3.   Generate mode constraints.
 4.   While no termination criterion is satisfied:
 5.     Call SAT solver to retrieve a model for the constraints.
 6.     If no model can be retrieved, terminate search (exhaustion).
 7.     Convert model into a (valid) candidate.
 8.     Evaluate candidate (using any evaluation function).
 9.     Update best-so-far candidate.
10.     Generate pruning constraints based on evaluation.
11.   Pick best candidate, remove cover (sequential covering).
```

Algorithm 1: NrSample's Search Algorithm.

In the following sections we focus on important details of SAT solving, search order, and maximum clause lengths (step 5), as well as efficient candidate evaluation (step 8).

3.1 Solving the Constraint Database

Note that all the constraints presented in Section 2 are in disjunctive form; in other words, they are propositional clauses. Each constraint represents a logical requirement for what needs to be satisfied, so our constraint database is a conjunction of clauses. Hence our database is in conjunctive normal form (CNF), which allows us to invoke many well studied SAT solvers without the need for CNF conversion (Russell et al., 1996).

For the general case, the Boolean satisfiability problem is known to be NP-complete (Cook, 1971). It may however not be clear that our SAT problems are NP-complete since the constraints contain regularities not assumed in the general case. By Definition 7, all mode declarations have at most one negative literal (that is, they are dual horn clauses). Satisfiability is hence decidable

in linear time (Dowling and Gallier, 1984). The pruning constraints, on the other hand, have all positive or all negative literals by Definition 11, leading to NP-completeness (Schaefer, 1978).

In practice, SAT has been extensively researched and high performance algorithms for quickly solving real world cases of the Boolean satisfiability problem exist. Chaff (Moskewicz et al., 2001) is a well known algorithm upon which many modern state-of-the-art SAT solvers are based, claiming to be able to solve problems with millions of variables.[4]

To ensure termination within a reasonable amount of time, we also provide a maximum number of literal assignments for the SAT solver. Our default value is 50000.

We will address the issue of SAT overhead from an empirical point of view in Section 4, but there is a point to be made about the NP-completeness of pruning constraints. The fact that finding a non-redundant candidate solution is NP-complete is not a flaw of our approach, but expresses the difficulty of constructing non-redundant solutions in ILP (with respect to pruning). Although not always true, insofar as this difficulty translates into low probabilities of randomly sampling a non-redundant candidate, algorithms that are indifferent about pruning constraints are no better off. In such cases, then, our algorithm would simply stop searching (due to the literal assignment limit, by default 50000), whereas other algorithms are likely to sample redundant candidates, therefore wasting time.[5]

3.2 Selection Strategies and Traversal Order

Since our approach relies on using a SAT solver to retrieve valid candidates, the search strategy—that is, the traversal order for candidate generation—is itself embedded into the SAT solver's model construction.

A *model* is a (propositional) assignment of truth values to all literals which satisfies all constraints. DPLL-based SAT solvers are a class of complete model builders (Russell et al., 1996; Davis et al., 1962). They construct models by first selecting an unassigned literal and assigning it either to true or false (according to some selection strategy), then propagating the effects of this assignment to all propositional clauses in the database. Propagation of an assignment $b_i = true$ works as follows: all clauses containing the literal b_i are removed (since they are now covered), and all clauses containing its negation $\neg b_i$ will have that literal removed (since that literal is not covered). When assigning $b_i = false$, the role of b_i and $\neg b_i$ are simply interchanged. Clauses that now only contain one literal—called *unit clauses*—indicate a definite assignment, so its effects are also propagated to all clauses.

When no more propagations are possible, either all literals have been assigned and we have a model, or we have reached a new choice point where any unassigned literal may be assigned true or false (again, this is handled by a selection strategy).

In what follows, we will assume that the DPLL-based algorithm follows the traditional technique of backtracking to the first literal that has not been tried both ways (that is, has not previously been assigned both to true and false). This is known as chronological backtracking; the case of non-chronological backtracking will be treated later.

Propagation may produce an empty clause, signifying a contradiction. This triggers a process of backtracking through the stack of assignments, searching for the most recent assigned literal that

4. This claim is made on Chaff's official webpage, `http://www.princeton.edu/~chaff/zchaff.html`. Visited on 2012-08-14.
5. The argument that DPLL-based SAT solvers can get stuck in entire subtrees with no solution, whereas the solution is readily available in a sibling branch, is not valid when random restarts are employed.

has not been tried with both assignments. This literal's assignment is then flipped, and we re-enter the process of propagating, backtracking, and selecting literals.

Freedom to pick any unassigned literal and assign it either true or false corresponds to choosing a literal from the bottom clause and deciding whether to include it or not (recall that a propositional literal b_i correspond to the ith first-order literal of the bottom clause). The restriction imposed by DPLL-based solvers comes from the fact that, if our choices of literals fail to satisfy the constraints, we may not change the assignments arbitrarily, but rather, must let the SAT solver flip all assignments (in reverse chronological order) that have not been tried both ways first.

Example 14 *Consider making the sequence of assignments* $(b_3 = true, b_1 = false, b_8 = true)$*, upon which the database is inconsistent (the empty clause was generated at the point where we assigned* $b_8 = true$*). This corresponds to choosing that the third and eighth literal of the bottom clause should be included in the candidate, whereas the first should not. The remaining literals have yet to be specified for inclusion/exclusion. If we had full freedom to choose any search strategy, we may for example want to start from scratch and exclude the third literal, not caring about the first and eighth. DPLL's backtracking algorithm prevents us from doing so however: it will first attempt to flip the last made assignment* $b_8 = true$ *into* $b_8 = false$*, thus producing the sequence* $(b_3 = true, b_1 = false, b_8 = false)$*. This makes it clear that we have no say in what the third, first, and eighth literal should be until the DPLL algorithm has backtracked to those choice points. Propagation may also force certain assignments, but this is desirable since it is this mechanism that excludes invalid candidates. After propagation, we are free to select any of the remaining unassigned literals in our next choice point, and assign it to either true or false.*

Example 14 is a simplification, as modern DPLL-based algorithms do not necessarily flip the last literal not tried both ways. Instead, they may go back further, to an older choice point and flip its literal, known as non-chronological backtracking or backjumping. This affects the predictability of the search (the example above is no longer valid), but it remains true that we have full freedom to select an assignment when, and only when, a choice point is reached. Chaff (Moskewicz et al., 2001) uses backjumping, but for a predictable search order, our algorithm uses chronological backtracking.

Some simple selection strategies include randomly assigning a literal, or always selecting the literal that occurs the most often in the database. Chaff introduced an efficient selection strategy known as VSIDS (Variable State Independent Decaying Sum) (Moskewicz et al., 2001). Which strategy would do best for ILP depends on the problem domain.

As we will see later, our benchmarks compare NrSample's performance against an algorithm that emulates its traversal path without using propositional constraints. Hence it must be reasonably easy for the emulation to also emulate the selection strategy without such constraints. This excludes VSIDS, as it explicitly depends on counting occurrences of propositional literals in the constraint database. Our selection strategy is to assign the literals in reverse order, to false first: $(b_n = false, b_{n-1} = false, \ldots, b_1 = false)$. This way, we always start with the top element, and backtracking starts flipping b_1 before b_2, b_2 before b_3, and so on.

Example 15 *If we assume that no backtracking occurs until the last assignment, we can visualize the sequence of candidates generated according to the aforementioned selection strategy (assigning literals to falsehood in reverse order). In bit string notation (0 signifies false and 1 true), with a bottom clause containing* 3 *literals, this sequence is:* $(000, 100, 010, 110, 001, 101, 011, 111)$*. The first candidate,* 000*, comes from assigning the literals to false in reverse order:* $(b_3 = 0, b_2 =$

$0, b_1 = 0$). *We then backtrack, flipping the last made assignment* $b_1 = 0$ *into* $b_1 = 1$. *This gives the assignment* $(b_3 = 0, b_2 = 0, b_1 = 1)$, *that is, the candidate* 100. *Next,* b_1 *has been tried both ways, so we remove the assignment.* b_2 *is now flipped, so we have the partial assignment* $(b_3 = 0, b_2 = 1)$. *Since we always start by assigning to falsehood, we then add* $b_1 = 0$, *obtaining the assignment* $(b_3 = 0, b_2 = 1, b_1 = 0)$, *and thus the candidate* 010. *This process continues until all candidates have been tried. Note that in practice, the assignments are unlikely to always reach the last literal before backtracking occurs due to the presence of constraints.*

Note that the restrictions imposed by DPLL-based SAT solvers are due to its backtracking mechanism, which is used to ensure completeness. It is also possible to use non-DPLL-based SAT solvers (Selman et al., 1995) with NrSample, for which no restrictions are imposed regarding literal selection strategy. However, care must be taken to ensure termination when no model is available: a timeout may be used when a solution cannot be found after a certain amount of time has elapsed or a maximum number of literal assignments tried. This ensures termination but not completeness.

That we can never have full control of search order traversal using our propositional constraints is clear, since the very reason for using them is to prevent certain candidates from being generated. However, it is in part possible to overcome limitations of DPLL-based backtracking by directing the search to desirable regions. This can be achieved by inserting a constraint to specify that region. When all candidates have been depleted, we negate this constraint, which forces the SAT solver to explore the complement region. This process can be done recursively, further partitioning subregions. We call such temporary constraints *regional constraints*.

Example 16 *Assume we want to start by exploring all candidates containing only a subset of the first 3 literals. The regional constraint is then* $\neg b_4 \wedge \neg b_5 \wedge \ldots \wedge \neg b_n$, *which translates into propositional clauses* $\{\neg b_4\}, \{\neg b_5\}, \ldots, \{\neg b_n\}$. *Once the search space has been exhausted, we negate this formula to obtain the complement region:* $b_4 \vee b_5 \vee \ldots \vee b_n$. *While exploring any of these two regions, we can apply the same principle to further partition regions into subregions.*

Care must be taken to treat regional constraints differently, as failure to satisfy them does not necessarily mean that no valid candidate exists: it only specifies that this subregion has been fully explored; what remains is now the complement region.

3.3 Maximum Clause Lengths

ILP systems give users the option to alter default parameters related to theory induction. For example, Progol allows users to set a limit on the number of iterations in the bottom clause construction, as well as on the maximum number of candidates to be evaluated (per bottom clause). These limits do not interfere with NrSample's constraint framework, since bottom clause construction is separated from search.

However, it is useful to also restrict the number of body literals allowed in any candidate solution during the search.[6] Although this restriction could be implemented as propositional constraints in NrSample, it is tedious to do so as each combination that is not allowed has to be specified explicitly. For example, with a bottom clause containing $n = 4$ body literals, the following formulae are needed to restrict all its candidates to at most $c = 2$ of those: $\{\neg b_1, \neg b_2, \neg b_3\} \wedge \{\neg b_1, \neg b_2, \neg b_4\} \wedge$

6. This is achieved using the query $set(c, N)$ in Progol and $set(clauselength, N)$ in Aleph.

$\{\neg b_1, \neg b_3, \neg b_4\} \wedge \{\neg b_2, \neg b_3, \neg b_4\}$. In general, we need to specify that no clause with $c+1$ literals is allowed, so we get $\binom{n}{c+1}$ constraints. A more efficient way to implement this restriction is to do so in the SAT solver itself. Since the Chaff algorithm works by assigning a truth value to a propositional variable and unit propagating the implied assignments, we keep track of the number of variables assigned true at all times. Whenever the limit is exceeded, backtracking occurs, ensuring that we never generate a solution with more than c positive assignments. As this is a non-standard feature of DPLL solvers, we have embedded our own implementation of the Chaff algorithm into NrSample.

Moreover, we generalize this technique so that our algorithm stores a mapping of which literals were generated from what mode declarations during bottom clause construction, and then keep track of how many times a mode declaration has been used during SAT solving. This enables us to specify upper bounds on the number of times a certain mode may be used in a candidate solution, a feature which is particularly useful to prevent large number of recursive literals in a predicate. This is implemented by using an extended mode declaration $modeb(Recall, Atom, Max\_Occurs)$ in which the third argument *Max_Occurs* specifies the maximum number of times a literal generated from this mode can occur in any candidate solution. Note that the bottom clause may still contain more occurrences, although not all of them may be used at the same time in a candidate. A search strategy based on this idea was proposed in Camacho (2000), where the number of occurrences of a predicate is progressively incremented.

3.4 Lexicographic Evaluation Function

The quality of a candidate is computed using an evaluation function. Different evaluation functions are possible. For example, if the number of positive and negative examples covered is P and N respectively, and L is the number of literals in the body of the candidate, Progol's A^* computes the fitness as $P-N-L-E$, where E is the fewest literals needed to get an input-output connected clause.[7] By default, Aleph uses the evaluation function P/T, where T is the total number of examples to be evaluated.

NrSample's default evaluation function assumes that our quality measure obeys a lexicographic ordering for $(P,-L)$. That is, $(P_1,-L_1) < (P_2,-L_2)$ if and only if $P_1 < P_2$ or $P_1 = P_2$ and $L_2 < L_1$. Intuitively, this states that (consistent) candidates are first compared by coverage, and only when they cover equally much are they compared by number of literals (fewer literals is better). We never insert an inconsistent candidate into the knowledge base, regardless of search algorithm. If no consistent candidate is found, we add the example itself to the knowledge base.

With respect to our quality measure, there are two observations that will reduce the number of examples evaluated.

First, consistency of a candidate is defined solely in terms of negative example coverage; there is hence no need to evaluate any positive examples to determine consistency. Since we never add an inconsistent candidate, there is no need to evaluate positive example coverage when the candidate is inconsistent. Thus all candidates should first be evaluated for negative coverage, and never for positive whenever they turn out to be inconsistent. Negative coverage evaluation can stop as soon as we detect one covered negative example.[8]

7. This corresponds to the number of positive assignments in a minimal model for the mode constraints.
8. More generally, with noise tolerance T, we stop after covering T negative examples. For $T = 0$ we have the noise free setting.

Second, we may safely abort positive coverage computation when there are not enough positive examples left for a candidate to beat the best-so-far. For example, if we have a best-so-far candidate that covers 20 out of 50 positive examples, and our candidate under evaluation covers 3 out of the first 34 examples, we may safely abort the last 16 examples since, even if they were all covered, it would still only amount to $3+16=19$ which is less than the already 20 previously covered. This optimization is possible since we need not consider clause lengths when two clauses have different coverage.

4. Experimental Results

Our benchmarks have two purposes.

First, we want to directly measure the effects of using propositional constraints in NrSample. Put differently, we would like to know whether producing, storing, and solving propositional constraints provides any real benefit over simply generating each candidate and then checking whether it conforms to the input-output specification and is logically non-redundant. To this end, we use an algorithm called *emulate_nrsample*, which searches for candidates in exactly the same order as NrSample, but without using constraints. Since both NrSample and emulate_nrsample traverse the search space in identical ways, comparing their execution time effectively measures the performance difference between solving the constraints and discarding invalid candidates.

Second, we want to measure NrSample with well established algorithms. Here we turn to the more general question of how useful NrSample is as an induction algorithm. To answer this, we compare NrSample against Progol's A^* and Aleph's enumeration search. For a fair comparison, all algorithms are implemented in our ILP system Atom, using the same libraries and backend.

4.1 The Search Algorithms

Progol's A^* is best-first search applied to ILP using the evaluation function $P-N-L-T$ where P and N are the number of positive and negative examples covered, respectively, L the number of body literals in the clause, and T the minimum number of literals needed to get an output instantiated candidate.

Enumeration search constructs the candidates of the search lattice level by level starting from the top. First, the top element is created, followed by all candidates with one body literal, then all candidates with two body literals, and so on. Seen as bit strings $(s_i)_1^n$, where 1s indicate that literal b_i from the bottom clause is used, we start with the candidate corresponding to bit string "11...00", the number of 1s corresponding to the current level as described above, and take previous permutations until we reach "00...11". In other words, in each layer we start by generating the candidate containing all leftmost literals of the bottom clause (as many as the depth we are considering), and cycle through all permutations until we reach the candidate that has all right-most literals.

Enumeration search uses the same evaluation function as NrSample, with all optimizations as described in Section 3. Progol's A^* cannot use any of these optimization as the heuristic needs to know the precise number of positive and negative examples covered in order to decide which nodes to expand.

The search strategy used by NrSample assigns right-most literals to false first (that is, in the sequence $b_n, b_{n-1}, \ldots, b_1$), so that the top element is explored first. It will then start flipping assignments in reverse order, so that b_1 will be assigned true first, then b_2, etc. For details, see Section 3.2, and, in particular, Example 15. We use no random restarts, making our algorithm deterministic.

emulate_nrsample generates each candidate in turn and then checks for mode and input-output conformance. When all invalid candidates have been discarded, the traversal order is the same as that of NrSample. Thus emulate_nrsample avoids the overhead of solving constraints, at the expense of potentially generating a large amount of unnecessary candidates.[9]

4.2 Test Problems and Data Sets

All comparisons are done using a set of well known ILP problems.

Three concept learning problems are used: ANIMALS, GRAMMAR, and TRAIN. These are taken from the data set in the Progol 4.4 distribution.[10]

ANIMALS is a classification problem: a list of animals is to be divided into mammals, fishes, reptiles, and birds. GRAMMAR presents examples of well formed and ill formed sentences, as well as a tiny dictionary of words grouped by word classes. The goal is to learn phrase structure rules. TRAIN is a binary classification problem: trains are either eastbound or westbound, depending on properties of their cars. The goal is to determine these dependencies.

Our remaining tests are program synthesis problems: MEMBER, SORTED, REVERSE, LENGTH, ADD, APPEND, SUBLIST.[11] MEMBER learns conventional Prolog list membership. SORTED is a unary predicate determining whether a list is sorted or not. REVERSE is true if and only if its second argument—a list—is a reversed form of its first (the relation is symmetric, but this is not exploited). LENGTH determines the number of elements in a list. The lists contain integers, which confuses the learning algorithms as it is not clear that the integer value of the list elements have nothing to do with the list length. This makes the problem significantly harder to solve. ADD defines formal addition in Peano arithmetic between two natural numbers. They are represented using the successor function (for example, integer 3 is represented as $s(s(s(0)))$. APPEND defines list concatenation. Finally, SUBLIST defines the ordered subset relation for lists: $sublist([],A)$, $sublist([A|B],[A|C]) \leftarrow sublist(B,C)$, $sublist(A,[B|C]) \leftarrow sublist(A,C)$.

Each of the program synthesis problems come in two variants. In the first variant, we use a set of predicates that is particularly well suited to each concept. For example, for APPEND, this would be operations to construct and deconstruct lists. Tables 8 and 9 in Appendix C shows the number of positive and negative examples, as well as mode declarations, for all problems. We refer to these problems—including the concept learning problems ANIMALS, GRAMMAR, and TRAIN—as the *small problems*.

In the second variant, we use a fixed set of predicates across *all* program synthesis problems. That is, we include mode declarations that allow for constructing and deconstructing lists, comparing/ordering integers, and performing elementary arithmetic on numbers. The precise definition of this primitive set of predicates is given in Appendix D. We refer to these problems as the *large problems*, and distinguish them from the small problems by using a prefix "L:". For example, MEMBER refers to the small problem and L:MEMBER to the large problem. As the primitive set is primarily

9. From a technical point of view, emulate_nrsample does not actually generate candidates as clauses, but rather, as a set representing which literals from the bottom clause each candidate is made up of. This is enough to check whether it is mode and input-output conformant. If the candidate is valid, the full candidate is generated and checked for coverage. This improves the performance of emulate_nrsample. The same optimization is used for enumeration. Progol's A^* does not check for input-output conformance of candidates, as it may need to expand invalid nodes to reach valid ones.
10. Available in the distribution of Progol 4.4 at `http://www.doc.ic.ac.uk/~shm/Software/progol4.4/progol4_4.tar.gz`. Visited on 2012-08-14.
11. Distributed with Atom's source code, see previous footnote.

intended for list and integer manipulation, the concept learning problems are not included in the large problems.

As the large problems use a fixed set of predicates—of which many predicates are intended for arithmetic computations, and thus functional in nature—we take the opportunity to test NrSample with functional constraints (*NrSFun*) against NrSample without functional constraints (*NrS*). Functional predicates are described in Section 2.5, and all functional predicates in the primitive set (see Appendix D) have the keyword "functional" in their modeb declaration. For *NrSFun*, this generates functional constraints, whereas *NrS* simply ignores the keyword.

The large amount of primitive predicates causes a combinatorial explosion of possibilities, thereby creating very large bottom clauses. We believe these large data sets better correspond to practical use of program synthesis, as the primitive predicates are general enough to induce all aforementioned concepts without any tweaking.

For each program synthesis problem, we generate 10 data sets. Each has its own (independently generated) positive and negative examples. We now describe how the examples in each of these data sets are generated.

All tests except ADD involve lists. Thus we need an algorithm to sample lists. We assume the lists hold a finite number of constants as elements. First, we note that the sample space is infinite, as lists can get arbitrarily long. Second, except for a finite subset, lists of increasing sizes necessarily must have diminishing probabilities.[12] This is also reasonable, since we expect simple observations (short lists) to be more common than elaborate ones. We first define the length L of a sampled list to be geometrically distributed with success probability 0.25: $P(L = k) = 0.75^k \cdot 0.25$. This makes sampling short lists more likely than long lists but puts no upper bound on the length of a list. For each of the L positions, we uniformly sample for a single digit constant ($\{0, 1, \ldots, 9\}$).

With our list generator, the positive examples are generated in the obvious way: for a positive example of MEMBER, sample a list and randomly pick an element of the list. For a negative example, sample a list not containing all domain elements and randomly pick an element not in the list. For APPEND, randomly sample two lists, and then append them to obtain the appended list. For a negative example, randomly sample three lists, and verify that the third is not obtained by appending the first and second. Also, we ensure no duplicate examples are generated.

For the ADD problem, we sample uniformly from $\{0, 1, 2, 3, 4\}$, providing the first $5 \cdot 5 = 25$ ground instances as positive examples. The reason for limiting formal addition to small examples is due to a depth limit of $h = 30$ when performing queries: each application of the successor function requires one call, so the largest number that can be handled by queries is 30. Such computational limits are necessary to make ILP problems tractable; all limits used in our benchmarks are listed in Section 4.3. L:ADD is different from ADD in that it does not represent numbers using the successor function but using lists of zeros. The length of the list is the number to be represented, that is, 3 is represented as $[0, 0, 0]$. This is because the primitive set in the large problems was primarily chosen with list manipulation in mind.

The concept learning data sets—ANIMALS, GRAMMAR, and TRAIN—are taken from Progol without modifications. For each of these three problems, we consider 10 different random orderings of the examples. Since the greedy sequential covering algorithm depends on the order of the positive examples, this affects the results.

12. Give the lists an enumeration and let p_n be the probability of sampling list n. $\sum p_n = 1$, which implies $p_n \to 0$.

4.3 Cross Validation

We consider two measures of the quality of an algorithm: how accurate the solution is, and the time it takes to generate a solution (execution time). Our benchmarks are performed using cross validation.

On the concept learning problems—ANIMALS, GRAMMAR, and TRAIN—which contain few positive examples, we perform leave-one-out experiments.

On the program synthesis problems, we use hold-out validation with 10 different data sets. Accuracy is computed as the fraction of all correct predictions among all predictions made.

For the sake of tractability, we also impose some computational limits to all benchmarked algorithms. During bottom clause construction, a maximum of $i = 3$ iterations are performed. For each bottom clause constructed, a maximum of $n = 1000$ (valid) candidates are explored. Candidates are restricted to a maximum clause length of $c = 4$. For each query, we use a recursion depth limit of $h = 30$ and maximum resolution limit of $r = 10000$. A time limit of $t = 600$ seconds is imposed on any induction. Upon reaching this time limit, the entire induction aborts, although post processing is still necessary to ensure all remaining examples are put back into the knowledge base. Except for the time limit, these restrictions are commonly used in Progol and Aleph (the values, of course, depend on the problem domain).

Progol A^* evaluates all candidates, regardless of mode declarations—this is necessary in order to choose the best node to expand in the search lattice. Aleph's enumeration explicitly states that the invalid candidates are to be ignored, as the following quotation from the Aleph homepage shows:

```
With these directives Aleph ensures that for any hypothesised clause
of the form H:- B1, B2, ..., Bc:

Input variables.
Any input variable of type T in a body literal Bi appears
as an output variable of type T in a body literal that appears before Bi,
 or appears as an input variable of type T in H.

Output variables.
Any output variable of type T in H appears as an output variable
of type T in Bi.
```

Without the time limit t, this algorithm may thus end up stuck in almost infinite generate-and-test loops when the number of valid candidates is small.

NrSample also needs a limit to ensure tractability while solving its propositional constraints: we set a limit of 50000 literal assignments per model constructed (equivalently: for each constructed candidate). When this limit is reached, the algorithm simply gives up the current bottom clause search, adds the example, and resumes sequential covering.

All benchmarks are performed on an Intel Core i7 (4×2.4 GHz) with 8 GB of RAM. All displayed numerical results are rounded to 3 decimals. For easier readability, trailing zeros are not shown (that is, we write "1" rather than "1.000").

Test	A_{NrS}	A_{Em}	A_{A^*}	A_E	A_{Em}/A_{NrS}	A_{A^*}/A_{NrS}	A_E/A_{NrS}
ANIMALS	0.9	0.9	0.952	0.971	1	1.058	1.079
GRAMMAR	0.981	0.981	0.952	1	1	0.970	1.019
TRAIN	1	1	0.9	1	1	0.9	1
MEMBER	0.983	0.983	0.935	0.983	1	0.951	1
SORTED	0.911	0.911	0.894	0.911	1	0.981	1
REVERSE	0.818	0.818	0.816	0.818	1	0.998	1
LENGTH	0.672	0.672	0.665	0.672	1	0.990	1
ADD	0.922	0.691	0.778	0.928	0.749	0.844	1.007
APPEND	0.804	0.411	0.783	0.590	0.511	0.974	0.734
SUBLIST	0.891	0.891	0.836	0.887	1	0.938	0.996

Table 1: Accuracy for Small Data Sets.

4.4 Results for Small Data Sets

Table 1 displays the accuracy of each algorithm as well as comparative ratios. A denotes Accuracy (defined in Section 4.3), with index NrS for NrSample, Em for emulate_nrsample, A^* for Progol's A^*, and E for enumeration. Since higher accuracy is better, the ratios compare favorably to NrSample when they are less than 1. Table 2 displays execution time—denoted T and measured in seconds—as well as comparative ratios. As lower execution time is better, a ratio larger than 1 is favorable to NrSample.

It is also interesting to know how often NrSample exhausts all valid candidates from search spaces. NrSample manages to exhaust all search spaces except REVERSE (95% exhausted), ADD (80%), and APPEND (60%). This suggests that—at least for NrSample—APPEND is the most difficult problem, followed by ADD and REVERSE.

As expected, NrSample has the same accuracy as emulate_nrsample on all small data sets except APPEND and ADD, where the latter times out. This is due to the fact that emulate_nrsample explores the same candidates as NrSample, provided no time out occurs. For the tests where emulate_nrsample did not time out, it sometimes completed the induction slightly faster than NrSample (ANIMALS, MEMBER, SORTED, LENGTH, and, most notably, TRAIN). On the tests with largest search spaces—APPEND, ADD, and REVERSE—NrSample performed substantially better than its emulated counterpart: 33, 4.5, and 9.4 times better, respectively.

As NrSample relies on input-output constraints, it is informative to divide the experimental results into concept learning—ANIMALS, GRAMMAR, and TRAIN—and program synthesis (all other problems).

On the concept learning problems, NrSample does not display any notable improvements over Progol A^* and enumeration. In particular, NrSample is substantially slower than both on the TRAIN problem: $1/0.07 \approx 14$ times slower than Progol's A^* although with better accuracy, and $1/0.016 \approx 63$ times slower than enumeration with same accuracy. On the other hand, NrSample is 63 times faster than Progol's A^* on GRAMMAR; indeed, it is the fastest algorithm on this problem, with comparable accuracy.

We note that accuracy on the concept learning problems is not always perfect (with any algorithm), despite their relative simplicity. This is because there are too few examples to always induce the correct definitions, even with leave-one-out. For example, in one instance of ANIMALS, all def-

Test	T_{NrS}	T_{Em}	T_{A^*}	T_E	T_{Em}/T_{NrS}	T_{A^*}/T_{NrS}	T_E/T_{NrS}
ANIMALS	0.021	0.018	0.057	0.018	0.857	2.714	0.857
GRAMMAR	0.018	0.021	1.133	0.021	1.167	62.944	1.167
TRAIN	2.575	1.438	0.18	0.04	0.558	0.070	0.016
MEMBER	0.163	0.156	28.637	0.156	0.957	175.687	0.957
SORTED	0.134	0.129	47.667	0.128	0.963	355.724	0.955
REVERSE	0.135	0.127	4.228	0.356	9.407	31.319	2.637
LENGTH	1.161	1.098	256.512	1.101	0.946	220.941	0.948
ADD	131.979	600.003	215.898	59.742	4.546	1.636	0.453
APPEND	18.088	600.019	43.082	600.027	33.172	2.382	33.173
SUBLIST	26.232	50.538	376.864	23.499	1.927	14.367	0.896

Table 2: Execution Time in seconds for Small Data Sets.

initions are correct except one, which is too general: $class(A, fish) \longleftarrow has\_legs(A, 0)$. The correct definition cannot be induced as the (only) example we need is being held out for validation.

Comparing NrSample against Progol A^* and enumeration on the program synthesis problems, we see that NrSample has better accuracy on all tests (except a tiny difference in favor of enumeration on ADD). It is also substantially faster than Progol's A^* on all problems, ranging from 1.6 to 356 times faster. It is slightly slower than enumeration on most tests: the exceptions are ADD, where enumeration is $1/0.453 = 2.2$ times faster, REVERSE, where NrSample is 2.6 times faster, and APPEND, where NrSample is 33 times faster.

4.5 Results for Large Data Sets

The large data sets test five algorithms: NrSample with functional constraints (*NrSFun*), NrSample without functional constraints (*NrS*), Emulated NrSample (*Em*), Progol's A^* (A^*), and enumeration E.

Table 3 shows accuracy on all problems, Table 4 shows accuracies compared with *NrSFun*. Since higher accuracy is better, lower values are in favor of *NrSFun*. Table 5 shows execution time, and Table 6 shows execution time compared with *NrSFun*. Since lower execution time is better, higher values are in favor of *NrSFun*.

The proportions of search spaces fully exhausted by *NrSFun* and *NrS* are given in Table 7. As pointed out earlier, NrSample detects exhaustion due to the unsatisfiability of its constraints, saving execution time. The results suggest that the most difficult problem in this regard is APPEND, followed by SUBLIST. SORTED and LENGTH are relatively difficult for *NrS*, which exhaust 60% and 70% respectively, whereas *NrsFun* fully exhaust all search spaces of both those problems.

As can be seen, NrSample with functional constraints induces with similar or better time performance compared to NrSample without functional constraints: on average 21 times faster. Notably, *NrSFun* is 11 times faster on MEMBER, 53 times faster on LENGTH, and 80 times faster on SORTED. All performance gains come without any accuracy penalty: accuracy is similar or better on all problems. Moreover, *NrSFun* is substantially more accurate on SUBLIST: 80% versus 48%.

Next, we note that emulate_nrsample times out on all data sets. As a result, it has worse execution time and accuracy compared to NrSample on all problems, demonstrating that the propositional framework is necessary for efficient constraint solving.

Test	A_{NrSFun}	A_{NrS}	A_{Em}	A_{A^*}	A_E
L:MEMBER	0.99	0.981	0.333	0.935	0.472
L:SORTED	0.834	0.827	0.027	0.397	0.72
L:REVERSE	0.555	0.555	0.349	0.562	0.411
L:LENGTH	0.665	0.663	0.335	0.413	0.366
L:ADD	0.83	0.808	0.676	0.784	0.83
L:APPEND	0.932	0.942	0.333	0.708	0.372
L:SUBLIST	0.803	0.478	0.352	0.826	0.517

Table 3: Accuracy for Large Data Sets.

Test	$A_{NrS}/A_{\text{NrSFun}}$	A_{Em}/A_{NrSFun}	$A_{A^*}/A_{\text{NrSFun}}$	A_E/A_{NrSFun}
L:MEMBER	0.991	0.336	0.944	0.477
L:SORTED	0.992	0.032	0.476	0.863
L:REVERSE	1	0.629	1.013	0.741
L:LENGTH	0.997	0.504	0.621	0.55
L:ADD	0.973	0.814	0.945	1
L:APPEND	1.011	0.357	0.76	0.399
L:SUBLIST	0.595	0.438	1.029	0.644

Table 4: Relative Accuracy for Large Data Sets.

Continuing, *NrSFun* is approximately as accurate or better than Progol A^* on all problems. *NrSFun* is substantially faster, with a speedup of between 7 and 45 times on all problems except APPEND and SUBLIST. On APPEND, A^* substantially sacrificed accuracy (71% versus 93%) to induce 8 times faster. Only on SUBLIST is the advantage uncontested, as A^* has both better execution time and slightly better accuracy (3.6 times faster with 83% versus 80% accuracy). On average, *NrSFun* is 18 times faster than A^*.

Finally, *NrSFun* has substantially better accuracy than enumeration on all problems except ADD, where they are tied. *NrSFun* also has substantially better performance, ranging from 1 to 1358 times faster. On average, *NrSFun* is 236 times faster than enumeration.

Test	T_{NrSFun}	T_{NrS}	T_{Em}	T_{A^*}	T_E
L:MEMBER	5.432	61.345	600.009	123.998	575.604
L:SORTED	4.964	397.797	600.012	109.236	251.13
L:REVERSE	0.442	0.43	600.016	3.258	600.02
L:LENGTH	4.427	232.445	600.01	197.314	571.642
L:ADD	8.843	25.73	600.004	256.356	20.817
L:APPEND	171.736	169.064	600.018	21.267	600.019
L:SUBLIST	584.989	600.031	600.012	160.673	600.017

Table 5: Execution Time in Seconds for Large Data Sets.

Test	$T_{NrS}/T_{\text{NrSFun}}$	T_{Em}/T_{NrSFun}	$T_{A^*}/T_{\text{NrSFun}}$	T_{E}/T_{NrSFun}
L:MEMBER	11.293	110.458	22.827	105.965
L:SORTED	80.136	120.873	22.006	50.59
L:REVERSE	0.973	1357.502	7.371	1357.511
L:LENGTH	52.506	135.534	44.571	129.126
L:ADD	2.91	67.851	28.99	2.354
L:APPEND	0.984	3.494	0.124	3.494
L:SUBLIST	1.026	1.026	0.275	1.026

Table 6: Relative Time Execution for Large Data Sets.

Test	*NrSFun*	*NrS*
L:MEMBER	0.9	0.85
L:SORTED	1	0.6
L:REVERSE	0.95	0.95
L:LENGTH	1	0.7
L:ADD	1	1
L:APPEND	0	0
L:SUBLIST	0.05	0

Table 7: Proportion Search Spaces Fully Exhausted.

5. Conclusions

We have provided a novel framework for non-redundant candidate construction in inductive logic programming (ILP), using propositional constraints relative to a bottom clause. In particular, we have treated the case of using search space pruning constraints, mode constraints (input-output constraints), and functional constraints, showing substantial speedups in program synthesis. Other algorithms embed pruning in some form, either implicitly through their search method—typically using refinement operators (Nienhuys-Cheng and de Wolf, 1997)—or explicitly, such as through memorization. However, they lack a mechanism to directly construct valid candidate solutions based on such constraints, leading to significant overhead in trial-and-error candidate generation.

We compared NrSample to Progol's A^* and Aleph's enumeration search. On the small program synthesis tests, NrSample outperformed both Progol A^* and enumeration on accuracy. Enumeration was marginally faster than NrSample on most tests; the exceptions are ADD, where enumeration was 2.2 times faster, REVERSE, where NrSample was 2.6 times faster, and APPEND, where NrSample was 33 times faster. NrSample was also substantially faster than A^*, ranging from 1.6 to 356 times faster.

On the large program synthesis tests, NrSample with functional constraints (*NrSFun*) always outperformed enumeration—ranging from 1 to 1358 times faster—as its naive search space traversal is often unable to find good solutions (it times out). *NrSFun* is on average 236 times faster, with substantially better accuracy on all large problems. Progol's A^* also has severe difficulties keeping up with NrSample in induction speed: *NrSFun* is on average 18 times faster than Progol's A^*, always with similar or better accuracy. Progol's A^* is only faster on two tests: APPEND and SUBLIST. However, on APPEND, Progol's A^* substantially trades accuracy for faster induction speed: accu-

racy is 71% versus 93%, for a performance gain of 8 times faster. SUBLIST is the only test in which it is a clear winner: it is 3.6 times faster with similar accuracy.

On the concept learning problems, NrSample does not seem to offer any advantage over more conventional algorithms. This is expected, as there are fewer input-output constraints to exploit. Therefore, NrSample's overhead of using a SAT solver is never compensated for. Input-output constraints can however occur in concept learning problems, and the TRAIN problem is an example as to how: it specifies that we may only attach cars in one direction, that is, from train name to its cars, not from cars to train name. With more such constraints, NrSample may offer an advantage. Moreover, it is possible to automatically generate mode declarations by inspecting the examples (McCreath and Sharma, 1995). This may create artificial but useful input-output constraints that speedup induction for arbitrary problems.

We also showed that NrSample's constraint mechanism is not easily replaced without a SAT solver: NrSample always has at least as good accuracy as its emulated counterpart, emulate_nrsample. In particular, emulate_nrsample is unable to keep up with NrSample on the large problems, consistently timing out on every data set.

Functional constraints (see Section 2.5) provide a significant performance advantage: *NrSFun* is always similar or faster than *NrS*, with similar or better accuracy. On average, *NrSFun* is 21 times faster than *NrS*.

We have shown that mode constraints are linear time solvable, whereas pruning constraints are NP-complete (see Section 3.1). Most SAT instances are—despite their NP-completeness—easy to solve in practice. As we have previously argued, the difficulty of finding non-redundant solutions is not limited to our algorithm, but rather, an inherent property of non-redundancy. Any algorithm not considering constraints may in such cases be unlikely to stumble upon a non-redundant solution.

We have shown how regional constraints—constraints used to direct the search to certain regions of the search space before others—allow for control over search order within our constraint framework (see Section 3.2).

Our constraint satisfaction approach is generalizable to problems with noise by simply modifying the definition of consistency so as to allow it to cover a user specified number of negative examples (instead of 0).

Our SAT solver is deterministic. In applications where this is undesirable, it is possible to use a non-deterministic selection strategy (for example, random literal assignment). Another source of randomness comes from setting a non-zero probability for random restarts. In both cases, pruning constraints still ensure that no redundant candidate is generated.

Acknowledgments

The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [Project No. CityU 124409].

Appendix A. Proof of Mode Constraint Correctness

In this appendix we prove Theorem 8, which establishes that the mode constraints reflect Definition 5.

First, we show that the mode constraints only generate mode conformant candidates (soundness).

Theorem 15 *Let F be a propositional formula for the mode constraints of B. Let C be a candidate from B. If F generates C, then C is mode conformant.*

Proof Assume C is not mode conformant. Let $P_B = l_1 \wedge \ldots \wedge l_n$ where $l_i = b_i$ or $l_i = \neg b_i$. We have two cases: (1) C is not input-instantiated, or (2) C is not output-instantiated.

(1) A literal b_i in C has an uninstantiated input variable v. However, B is input-instantiated by Lemma 6, so $v \in I_h$, or $v \in O_k$ for some $k < i$ and b_k is not a body literal of C (since otherwise v would be instantiated). Now, $v \in I_h$ is impossible because then C would have v instantiated by its head input, as C and B have the same head. Let $k_1, \ldots, k_s$ be the indices for body literals in B for which $v \in O_{k_s}$, $k_s < i$. By Definition 7, F contains a clause $c = \{\neg b_i, b_{k_1}, \ldots, b_{k_s}\}$. Since C contains b_i but none of the preceding b_{k_j}, the model M_C for C has $M_C(b_i) = true$ and $M_C(b_{k_j}) = false$ for all b_{k_j}, $j = 1, \ldots, s$, so M_C does not satisfy c. Hence M_C is not a model for F.

(2) The head of C has an uninstantiated output variable v. Partition $b_1, \ldots, b_n$ into two sets: B_v for the b_k's that satisfy $v \in O_k$, and $B_{\neg v}$ for the rest. All literals of P_C are in $B_{\neg v}$, so P_C is a conjunction of literals where each literal of B_v is negative. Hence for the model M_C for P_C, $M_C(b_i) = false$ for all $b_i \in B_v$. But by Definition 7, F contains the clause $\bigvee_{b_i \in B_v} b_i$, so M_C is not a model for F. ■

Next, we show that the mode constraints can generate any mode conformant candidate (completeness).

Theorem 16 *Let F be a propositional formula for the mode constraints of B. Let C be a candidate from B. If C is mode conformant, then F generates C.*

Proof Assume C is mode conformant and let M_C be the model for $P_C = l_{s_1} \wedge \ldots \wedge l_{s_k}$. Then $M_C(l_{s_i}) = true$ for all $i = 1, \ldots, k$. Let f be a clause in F. Either f has the form $\{\neg b_n, b_{x_1}, b_{x_2}, \ldots\}$ or $\{b_{x_1}, b_{x_2}, \ldots\}$.

The first form appears when b_n has an input variable v that is not an output of B's head. b_{x_j} are the literals preceding b_n ($x_j < n$) in which v appears as output. If b_n does not appear in C, $M_C(b_n) = false$ and the clause is satisfied. If b_n appears in C, we note that since C is mode conformant, v appears in a previous body literal or the head. But it cannot appear in the head of C, since it is the same as the head of B. So v must be the output of a literal in C that appears before b_n, that is, one of the b_{x_j}. Hence $M_C(b_{x_j}) = true$ for this particular j, and the clause is satisfied.

The second form appears when the head of B has an output variable v. If the clause is empty, B has no body literal that outputs v, so no candidate from B is mode conformant either, contradicting our assumption that C is mode conformant. Now v appears in the head of C, again because it is the same as the head of B. Let $b_{x_1}, \ldots, b_{x_k}$ be the literals of the non-empty clause. Since C is mode conformant, v appears as output in a literal b_{x_j}, $j = 1, \ldots, k$, and hence $M_C(b_{x_j}) = true$. Therefore the clause is satisfied. ■

Appendix B. Proof of Pruning Constraint Correctness

In this appendix, we prove Theorem 12, establishing the correctness of NrSample's pruning constraints.

The next lemma will be needed in our correctness proof.

Lemma 17 *Let C and D be candidates from B. If $C \subseteq D$, all the negative literals of P_D occur in P_C.*

Proof If $C \subseteq D$, all the body literals of C occur in D, so $b_j \in P_C \Longrightarrow b_j \in P_D$. Let $\neg b_i \in P_D$. If $b_i \in P_C$, then $b_i \in P_D$, a contradiction. Since P_C contains all literals of $b_1, \ldots, b_n$, we have $\neg b_i \in P_C$. ■

The following theorem shows that $\neg P_{\uparrow C}$ and $\neg P_{\downarrow C}$ block the intended regions and no more.

Theorem 18 Let C be a candidate from B.

1. $\neg P_{\uparrow C}$ generates G if and only if G is not a generalization of C.
2. $\neg P_{\downarrow C}$ generates S if and only if S is not a specialization of C.

Proof (1) Assume $G \subseteq C$. If P_C has no negative literal, $\neg P_{\uparrow C} = false$, and the claim follows trivially. So assume P_C has negative literals $\bar{c_1}, \ldots \bar{c_k}$. Since $G \subseteq C$, $\bar{c_1}, \ldots \bar{c_k}$ are negative literals in P_G by Lemma 17. The model for G thus has $M_G(c_i) = false$ for all $i = 1, \ldots, k$. But $\neg P_{\uparrow C} = \{c_1, \ldots, c_k\}$, so it is false in M_G. For the converse, assume $G \not\subseteq C$. Then there is a positive literal $b \in P_G$ with $\neg b \in P_C$. Let M_G be the model for P_G: for all $g_i \in P_G$, $M_G(g_i) = true$, and for all negative $\bar{g_j} \in P_G$, $M_G(g_j) = false$. Note that $M_G(b) = true$. Since $\neg P_{\uparrow C}$ is a clause containing b , it is true under M_G.

(2) Assume $C \subseteq S$. If P_C has no positive literal, $\neg P_{\downarrow C} = false$, and the claim follows trivially. So assume P_C has positive literals $c_1, \ldots, c_k$. Assume M_S is a model for P_S. Since $C \subseteq S$, $c_1, \ldots, c_k$ are also in P_S, so $M_S(c_i) = true$ for all $i = 1, \ldots, k$. But $\neg P_{\downarrow C} = \{\neg c_1, \ldots, \neg c_k\}$, so it is false in M_S. For the converse, assume $C \not\subseteq S$. Then there is a positive literal $b \in P_C$ with $\neg b \in P_S$. Let M_S be the model for P_S: for all positive $s_i \in P_S$, $M_S(s_i) = true$, and for each negative $\neg s_j \in P_S$, $M_S(s_j) = false$. Note that $M_S(b) = false$. Since $\neg P_{\downarrow C}$ contains $\neg b$, M_S is a model for $\neg P_{\downarrow C}$. ■

Appendix C. Small Data Sets Details

Table 8 and 9 show the number of examples and mode declarations used in concept learning and program synthesis problems, respectively. Note that the number of examples in each data set is not necessarily indicative of complexity. For example, TRAIN has only 5 positive and 5 negative examples, but is harder to learn than Animals and Grammar. The complexity of a problem depends mainly on the mode declarations used for the body, as they give the set of all possible predicates to be used when constructing the bottom clause.

Appendix D. Primitive Set of Predicates

What follows are the definitions of the primitive set used in the large data set experiments (those prefixed by "L:").

```
:- modeb(1,nil(+list))?
:- modeb(1,+list = [-nonlist|-list],[functional])?
:- modeb(1,-list = [+nonlist|+list],[functional])?
:- modeb(1,listify(+nonlist,-list),[functional])?

listify(X,[X]).
```

Data Set	P	N	Mode Declarations
animal	16	42	$modeh(1, class(+animal, \#class))$ $modeb(1, has\_milk(+animal))$ $modeb(1, has\_gills(+animal))$ $modeb(1, has\_covering(+animal, \#covering))$ $modeb(1, has\_legs(+animal, \#nat))$ $modeb(1, homeothermic(+animal))$ $modeb(1, has\_eggs(+animal))$ $modeb(1, not\ has\_milk(+animal))$ $modeb(1, not\ has\_gills(+animal))$ $modeb(*, habitat(+animal, \#habitat))$ $modeb(1, class(+animal, \#class))$
grammar	14	7	$modeh(1, s(+wlist, -wlist))$ $modeb(1, det(+wlist, -wlist))$ $modeb(*, np(+wlist, -wlist))$ $modeb(*, vp(+wlist, -wlist))$ $modeb(1, prep(+wlist, -wlist))$ $modeb(1, noun(+wlist, -wlist))$ $modeb(1, verb(+wlist, -wlist))$
train	5	5	$modeh(1, eastbound(+train))$ $modeb(100, has\_car(+train, -car))$ $modeb(1, notopen(+car))$ $modeb(1, notlong(+car))$ $modeb(1, long(+car))$ $modeb(1, open(+car))$ $modeb(1, double(+car))$ $modeb(1, jagged(+car))$ $modeb(1, shape(+car, -shape))$ $modeb(1, load(+car, -shape, -int1))$ $modeb(1, wheels(+car, -int1))$ $modeb(1, infront(+car, -car))$

Table 8: Concept Learning Data Sets.

Data Set	P	N	Mode Declarations
member	100	50	$modeh(*, member(+const, +clist))$ $modeb(1, +any = \#any)$ $modeb(1, +clist = [-const \mid -clist])$ $modeb(*, member(+const, +clist))$
sorted	100	50	$modeh(1, sorted(+clist))$ $modeb(1, +const =< +const)$ $modeb(1, +clist = [-const \mid -clist])$ $modeb(1, +clist = [])$ $modeb(1, sorted(+clist))$
reverse	100	50	$modeh(1, reverse(+clist, -clist))$ $modeb(1, reverse(+clist, -clist))$ $modeb(1, +clist = [-const \mid -clist])$ $modeb(1, +clist = [])$ $modeb(1, append(+clist, [+const], -clist))$
length	100	50	$modeh(1, length(+clist, +int))$ $modeh(1, length(+clist, -int))$ $modeb(1, -intis + int + 1)$ $modeb(1, +intis\#int)$ $modeb(1, length(+clist, -int))$ $modeb(1, +int = 0)$ $modeb(1, +clist = [])$ $modeb(1, +clist = [-const \mid -clist])$
add	25	50	$modeh(*, add(+snum, +snum, -snum))$ $modeb(1, +snum = 0)$ $modeb(1, -snum = +snum)$ $modeb(1, dec(+snum, -snum))$ $modeb(1, inc(+snum, -snum))$ $modeb(1, add(+snum, +snum, -snum))$
append	100	50	$modeh(1, append(+list, +list, -list))$ $modeb(1, +list = [])$ $modeb(1, +const = +const)$ $modeb(1, +list = [-const \mid -list])$ $modeb(1, -list = [+const \mid +list])$ $modeb(1, append(+list, +list, -list))$
sublist	100	50	$modeh(1, sublist(+clist, +clist))$ $modeb(1, +clist = [-const \mid -clist])$ $modeb(1, +clist = [])$ $modeb(1, sublist(+clist, +clist))$

Table 9: Program Synthesis Data Sets.

```
nil([]).
nonlist(X) :- constant(X), X \= [].
list([]).
list([H|T]) :- nonlist(H), list(T).

:- modeb(1,inc(+number,-number),[functional])?
:- modeb(1,-number is +number + +number,[functional])?
:- modeb(1,neg(+number,-number),[functional])?
:- modeb(1,inv(+number,-number),[functional])?
:- _ is X+Y prevents _ is Y+X?
:- prevent neg(X,X)?
:- prevent inv(X,X)?

inc(X,Y) :- Y is X+1.
neg(X,Y) :- Y is -X.
inv(X,Y) :- Y is 1/X.

:- modeb(1,+number =:= +number)?
:- modeb(1,+number =\= +number)?
:- modeb(1,+number < +number)?
:- modeb(1,+number =< +number)?
:- prevent X =:= X? % reflexivity
:- X =:= Y prevents Y =:= X? % symmetry
:- prevent X =\= X?
:- X =\= Y prevents Y =\= X?
:- prevent X =< X?
```

We explain the operators prevent/1 and prevents/2 with examples. They are used during bottom clause construction, and are thus not specific to our benchmarked algorithms. Both operators are intended to prevent certain ground truths from occurring in the bottom clause (or their corresponding lifted instances). Example 17 illustrates its use. Similar ideas can be found in Fonseca et al. (2004).

Example 17 *The clause 'prevent $p(X,X)$' ensures that no bottom clause of the form $p(X,X)$ occurs, where matching is done so that X unifies with ground terms. So $p(a,b)$ is not allowed with this rule, but $p(a,a)$ is. The instance prevent $X =< X$ simply prevents trivial comparisons.*

As a matter of technicality, when using lifted bottom clauses, that is, with mode declarations, variables in the bottom clause must be treated as ground terms. For example, the prevent rule $X = X$ should not match with bottom clause literal $Y = 0$, as what we have in mind is to prevent trivial unifications of identical terms. By treating unlifted bottom clause literals (that is, ground truths), this problem does not arise.

Example 18 *With the rule 'prevent $p(X,X)$' from Example 17, the literal $p(A,B)$ is fails to match, since it is first grounded to $p(c_A,c_B)$, and unification then fails. Thus the literal $p(A,B)$ will not be prevented from appearing. On the other hand, $p(A,A)$ will be prevented, since it is treated as $p(c_A,c_A)$.*

Example 19 *The clause '$p(X,Y)$ prevents $q(Y,X)$' ensures that the literal $q(Y,X)$ does not occur* if *$p(X,Y)$ occurs as a previous literal in the bottom clause. Again, unification is used for matching, so that repeated variable occurrences matter. The instance used in our primitive set:*

```
_ is X+Y prevents _ is Y+X
```

simply exploits the commutativity of addition: if we have $X+Y$ (we do not care about the output variable), we do not need the addition $Y+X$ in the bottom clause.

References

Patrick Blackburn, Johan Bos, and Kristina Striegnitz. *Learn Prolog Now!* College Publications, 2006.

Rui Camacho. *Inducing Models of Human Control Skills using Machine Learning Algorithms*. PhD thesis, SciVerse, Pavel Brazdil, 2000.

Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM. doi: 10.1145/800157.805047. URL http://doi.acm.org/10.1145/800157.805047.

Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5:394–397, July 1962. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/368273.368557. URL http://doi.acm.org/10.1145/368273.368557.

Luc De Raedt and Jan Ramon. Condensed representations for inductive logic programming. In *Proceedings of the 9th International Conference on Principles of Knowledge Representation and Reasoning*, pages 438–446. AAAI Press, 2004.

William F. Dowling and Jean H. Gallier. Linear-time algorithms for testing the satisfiability of propositional horn formulae. *Journal of Logic Programming*, 1(3):267–284, 1984.

Pierre Flener and Serap Yılmaz. Inductive synthesis of recursive logic programs: Achievements and prospects. *Journal of Logic Programming*, 41:141–195, 1999.

Pierre Flener, Kung-Kiu Lau, Mario Ornaghi, and Julian Richardson. An abstract formalization of correct schemas for program synthesis. *Journal of Symbolic Computation*, 30(1):93–127, 2000.

Nuno A. Fonseca, Vítor Santos Costa, Fernando M. A. Silva, and Rui Camacho. On avoiding redundancy in inductive logic programming. In *ILP*, pages 132–146, 2004.

Jérôme Maloberti and Michèle Sebag. Fast theta-subsumption with constraint satisfaction algorithms. *Machine Learning*, 55(2):137–174, May 2004. ISSN 0885-6125. doi: 10.1023/B:MACH.0000023150.80092.40. URL http://dx.doi.org/10.1023/B:MACH.0000023150.80092.40.

Eric McCreath and Arun Sharma. Extraction of meta-knowledge to restrict the hypothesis space for ILP systems. In *Proceedings of the Eighth Australian Joint Conference on Artificial Intelligence*, pages 75–82. World Scientific, 1995.

Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: engineering an efficient SAT solver. In *Proceedings of the 38th Annual Design Automation Conference*, DAC '01, pages 530–535, New York, NY, USA, 2001. ACM. ISBN 1-58113-297-2. doi: http://doi.acm.org/10.1145/378239.379017. URL http://doi.acm.org/10.1145/378239.379017.

Stephen Muggleton and Alireza Tamaddoni-Nezhad. QG/GA: a stochastic search for Progol. *Machine Learning*, 70:121–133, March 2008. ISSN 0885-6125. doi: 10.1007/s10994-007-5029-3. URL http://dl.acm.org/citation.cfm?id=1331425.1331452.

Stephen Muggleton, Luc De Raedt, David Poole, Ivan Bratko, Peter A. Flach, Katsumi Inoue, and Ashwin Srinivasan. ILP turns 20 - biography and future challenges. *Machine Learning*, 86(1): 3–23, 2012.

Stephen H. Muggleton. Inverse Entailment and Progol. *New Generation Computing*, 13:245–286, 1995. URL http://www.doc.ic.ac.uk/~shm/Papers/InvEnt.ps.gz.

Shan-Hwei Nienhuys-Cheng and Ronald de Wolf. *Foundations of Inductive Logic Programming*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997. ISBN 3540629270.

Gordon D. Plotkin. A note on inductive generalization. *Machine Intelligence*, 5:153–163, 1970.

Gordon D. Plotkin. A Further Note on Inductive Generalization. *Machine Intelligence*, 6:101–124, 1971.

Stuart J. Russell, Peter Norvig, John F. Candy, Jitendra M. Malik, and Douglas D. Edwards. *Artificial intelligence: a modern approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996. ISBN 0-13-103805-2.

Thomas J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, STOC '78, pages 216–226, New York, NY, USA, 1978. ACM. doi: 10.1145/800133.804350. URL http://doi.acm.org/10.1145/800133.804350.

Michèle Sebag and Céline Rouveirol. Constraint inductive logic programming. In Luc De Raedt, editor, *Advances in ILP*. IOS Press, 1996.

Bart Selman, Henry Kautz, and Bram Cohen. Local search strategies for satisfiability testing. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 521–532, 1995.

Mathieu Serrurier, Henri Prade, and Gilles Richard. A Simulated Annealing Framework for ILP. In Rui Camacho, Ross D. King, and Ashwin Srinivasan, editors, *ILP*, volume 3194 of *Lecture Notes in Computer Science*, pages 288–304. Springer, 2004. ISBN 3-540-22941-8.

Ashwin Srinivasan. *The Aleph Manual*, 2001. URL http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/. http://web.comlab.ox.ac.uk/oucl/research/ areas/machlearn/Aleph/.

Leon Sterling and Ehud Shapiro. *The art of Prolog (2nd ed.): advanced programming techniques*. MIT Press, Cambridge, MA, USA, 1994. ISBN 0-262-19338-8.

Alireza Tamaddoni-Nezhad and Stephen Muggleton. The lattice structure and refinement operators for the hypothesis space bounded by a bottom clause. *Machine Learning*, 76(1):37–72, July 2009. ISSN 0885-6125. doi: 10.1007/s10994-009-5117-7. URL http://dx.doi.org/10.1007/s10994-009-5117-7.

Journal of Machine Learning Research 14 (2013) 3683-3719 Submitted 3/13; Revised 10/13; Published 12/13

How to Solve Classification and Regression Problems on High-Dimensional Data with a Supervised Extension of Slow Feature Analysis

Alberto N. Escalante-B. ALBERTO.ESCALANTE@INI.RUB.DE
Laurenz Wiskott LAURENZ.WISKOTT@INI.RUB.DE
Institut für Neuroinformatik
Ruhr-Universität Bochum
Bochum D-44801, Germany

Editor: David Dunson

Abstract

Supervised learning from high-dimensional data, for example, multimedia data, is a challenging task. We propose an extension of slow feature analysis (SFA) for *supervised* dimensionality reduction called graph-based SFA (GSFA). The algorithm extracts a label-predictive low-dimensional set of features that can be post-processed by typical supervised algorithms to generate the final label or class estimation. GSFA is trained with a so-called training graph, in which the vertices are the samples and the edges represent similarities of the corresponding labels. A new weighted SFA optimization problem is introduced, generalizing the notion of slowness from sequences of samples to such training graphs. We show that GSFA computes an optimal solution to this problem in the considered function space and propose several types of training graphs. For classification, the most straightforward graph yields features equivalent to those of (nonlinear) Fisher discriminant analysis. Emphasis is on regression, where four different graphs were evaluated experimentally with a subproblem of face detection on photographs. The method proposed is promising particularly when linear models are insufficient as well as when feature selection is difficult.

Keywords: slow feature analysis, feature extraction, classification, regression, pattern recognition, training graphs, nonlinear dimensionality reduction, supervised learning, implicitly supervised, high-dimensional data, image analysis

1. Introduction

Supervised learning from high-dimensional data has important applications in areas such as multimedia processing, human-computer interfaces, industrial quality control, speech processing, robotics, bioinformatics, image understanding, and medicine. Despite constant improvements in computational resources and learning algorithms, supervised processing, for example, for regression or classification, of high-dimensional data is still a challenge largely due to insufficient data and several phenomena referred to as the curse of dimensionality. This limits the practical applicability of supervised learning.

Unsupervised dimensionality reduction, including algorithms such as principal component analysis (PCA) or locality preserving projections (LPP, He and Niyogi, 2003), can be used to attenuate these problems. After dimensionality reduction, typical supervised learning algorithms can be applied. Frequent benefits include a lower computational cost and better robustness against overfitting.

However, since the final goal is to solve a supervised learning problem, this approach is inherently suboptimal.

Supervised dimensionality reduction is more appropriate in this case. Its goal is to compute a low-dimensional set of features from the high-dimensional input samples that contains predictive information about the labels (Rish et al., 2008). One advantage is that dimensions irrelevant for the label estimation can be discarded, resulting in a more compact representation and more accurate label estimations. Different supervised algorithms can then be applied to the low-dimensional data. A widely known algorithm for supervised dimensionality reduction is Fisher discriminant analysis (FDA) (Fisher, 1936). Sugiyama (2006) proposed local FDA (LFDA), an adaptation of FDA with a discriminative objective function that also preserves the local structure of the input data. Later, Sugiyama et al. (2010) proposed semi-supervised LFDA (SELF) bridging LFDA and PCA and allowing the combination of labeled and unlabeled data. Tang and Zhong (2007) introduced pairwise constraints-guided feature projection (PCGFP), where two types of constraints are allowed. Must-link constraints denote that a pair of samples should be mapped closely in the low-dimensional space, while cannot-link constraints require that the samples are mapped far apart. Later, Zhang et al. (2007) proposed semi-supervised dimensionality reduction (SSDR), which is similar to PCGFP and also supports semi-supervised learning.

Slow feature analysis (SFA) (Wiskott, 1998; Wiskott and Sejnowski, 2002) is an unsupervised learning algorithm inspired by the visual system and based on the slowness principle. SFA has been applied to classification in various ways. Franzius et al. (2008) extracted the identity of animated fish invariant to pose (including a rotation angle and the fish position) with SFA. A long sequence of fish images was rendered from 3D models in which the pose of the fish changed following a Brownian motion, and in which the probability of randomly changing the fish identity was relatively small, making identity a feature that changes slowly. This result confirms that SFA is capable of extracting categorical information. Klampfl and Maass (2010) introduced a particular Markov chain to generate a sequence used to train SFA for classification. The transition probability between samples from different object identities was proportional to a small parameter a. The authors showed that in the limit $a \to 0$ (i.e., only intra-class transitions), the features learned by SFA are equivalent to the features learned by Fisher discriminant analysis (FDA). The equivalence of the discrimination capability of SFA and FDA in some setups was already known (compare Berkes, 2005a, and Berkes, 2005b) but had not been rigorously shown before. In the two papers by Berkes, hand-written digits from the MNIST database were recognized. Several mini-sequences of two samples from the same digit were used to train SFA. The same approach was also applied more recently to human gesture recognition by Koch et al. (2010) and a similar approach to monocular road segmentation by Kuhnl et al. (2011). Zhang and Tao (2012) proposed an elaborate system for human action recognition, in which the difference between delta values of different training signals was amplified and used for discrimination.

SFA has been used to solve regression problems as well. Franzius et al. (2008) used standard SFA to learn the position of animated fish from images with homogeneous background. The same training sequence used for learning fish identities was used, thus the fish position changed continuously over time. However, a different supervised post-processing step was employed consisting of linear regression coupled with a nonlinear transformation.

In this article, we introduce a supervised extension of SFA called graph-based SFA (GSFA) specifically designed for supervised dimensionality reduction. We show that GSFA computes the slowest features possible according to the GSFA optimization problem, a weighted extension of the

SFA problem, generalizing the concept of signal slowness. The objective function of the GSFA problem is a weighted sum of squared output differences and is therefore similar to the underlying objective functions of, for example, FDA, LFDA, SELF, PCGFP, and SSDR. However, in general the optimization problem solved by GSFA differs at least in one of the following elements: a) the concrete coefficients of the objective function, b) the constraints, or c) the feature space considered. Although nonlinear or kernelized versions of the algorithms above can be defined, one has to overcome the difficulty of finding a good nonlinearity or kernel. In contrast, SFA (and GSFA) was conceived from the beginning as a nonlinear algorithm without resorting to kernels (although there exist versions with a kernel: Bray and Martinez, 2003; Vollgraf and Obermayer, 2006; Böhmer et al., 2012), with linear SFA being just a less used special case. Another difference to various algorithms above is that SFA (and GSFA) does not explicitly attempt to preserve the spatial structure of the input data. Instead, it preserves the similarity structure provided, which leaves room for better optimization towards the labels.

Besides the similarities and differences outlined above, GSFA is strongly connected to some algorithms in specific cases. For instance, features equivalent to those of FDA can be obtained if a particular training graph is given to GSFA. There is also a close relation between SFA and Laplacian eigenmaps (LE), which has been studied by Sprekeler (2011). GSFA and LE basically have the same objective function, but in general GSFA uses different edge-weight (adjacency) matrices, has different normalization constraints, supports node-weights, and uses function spaces.

There is also a strong connection between GSFA and LPP. In Section 7.1 we show how to use GSFA to extract LPP features and *vice versa*. This is a remarkable connection because GSFA and LPP originate from different backgrounds and are typically used for related but different goals. Generalized SFA (Sprekeler, 2011; Rehn, 2013), being basically LPP on nonlinearly expanded data, is also closely connected to GSFA.

One advantage of GSFA over many algorithms for supervised dimensionality reduction is that it is designed for both classification and regression (using appropriate training graphs), whereas other algorithms typically focus on classification only.

Given a large number of high-dimensional labeled data, supervised learning algorithms can often not be applied due to prohibitive computational requirements. In such cases we propose the following general scheme based on GSFA/SFA, illustrated in Figure 1 (left):

1. Transform the labeled data to structured data, where the label information is implicitly encoded in the connections between the data points (samples). This permits using unsupervised learning algorithms, such as SFA, or its extension GSFA.

2. Use hierarchical processing to reduce the dimensionality, resulting in low-dimensional data with component similarities strongly dependent on the graph connectivity. Since the label information is encoded in the graph connectivity, the low-dimensional data are highly predictive of the labels. Hierarchical processing (Section 2.4) is an efficient divide-and-conquer approach for high-dimensional data with SFA and GSFA.

3. Convert the (low-dimensional) data back to labeled data by combining the low-dimensional data points with the original labels or classes. This now constitutes a data set suitable for standard supervised learning methods, because the dimensionality has become manageable.

4. Use standard supervised learning methods on the low-dimensional labeled data to estimate the labels. The unsupervised hierarchical network plus the supervised direct method together constitute the classifier or regression architecture.

In the case of GSFA, the structured training data is called training graph, a weighted graph that has vertices representing the samples, node weights specifying *a priori* sample probabilities, and edge weights indicating desired output similarities, as derived from the labels. Details are given in Section 3. This structure permits us to extend SFA to extract features from the data points that tend to reflect similarity relationships between their labels without the need to reproduce the labels themselves. A concrete example of the application of the method to a regression problem is illustrated in Figure 2. Various important advantages of GSFA are inherited from SFA:

- It allows hierarchical processing, which has various remarkable properties, as described in Section 2.4. One of them, illustrated in Figure 1 (right), is that the local application of SFA/GSFA to lower-dimensional data chunks typically results in less overfitting than non-hierarchical SFA/GSFA.
- SFA has a complexity of $O(N)$ in the number of samples N and $O(I^3)$ in the number of dimensions I (possibly after a nonlinear expansion). Hierarchical processing greatly reduces the latter complexity down to $O(I)$. In practice, processing 100,000 samples of 10,000-dimensional input data can be done in less than three hours by using hierarchical SFA/GSFA without resorting to parallelization or GPU computing.
- Typically no expensive parameter search is required. The SFA and GSFA algorithms themselves are almost parameter free. Only the nonlinear expansion has to be defined. In hierarchical SFA, the structure of the network has several parameters, but the choice is usually not critical.

In the next sections, we first recall the standard SFA optimization problem and algorithm. Then, we introduce the GSFA optimization problem, which incorporates the information contained in a training graph, and propose the GSFA algorithm, which solves this optimization problem. We recall how classification problems have been addressed with SFA and propose a training graph for doing this task with GSFA. Afterwards, we propose various graph structures for regression problems offering great computational efficiency and good accuracy. Thereafter, we experimentally evaluate and compare the performance of four training graphs to other common supervised methods (e.g., PCA+SVM) w.r.t. a particular regression problem closely related to face detection using real photographs. A discussion section concludes the article.

2. Standard SFA

In this section, we begin by introducing the slowness principle, which has inspired SFA. Afterwards, we recall the SFA optimization problem and the algorithm itself. We conclude the section with a brief introduction to hierarchical processing with SFA.

2.1 The Slowness Principle and SFA

Perception plays a crucial role in the interaction of animals or humans with their environment. Although processing of sensory information appears to be done straightforwardly by the nervous

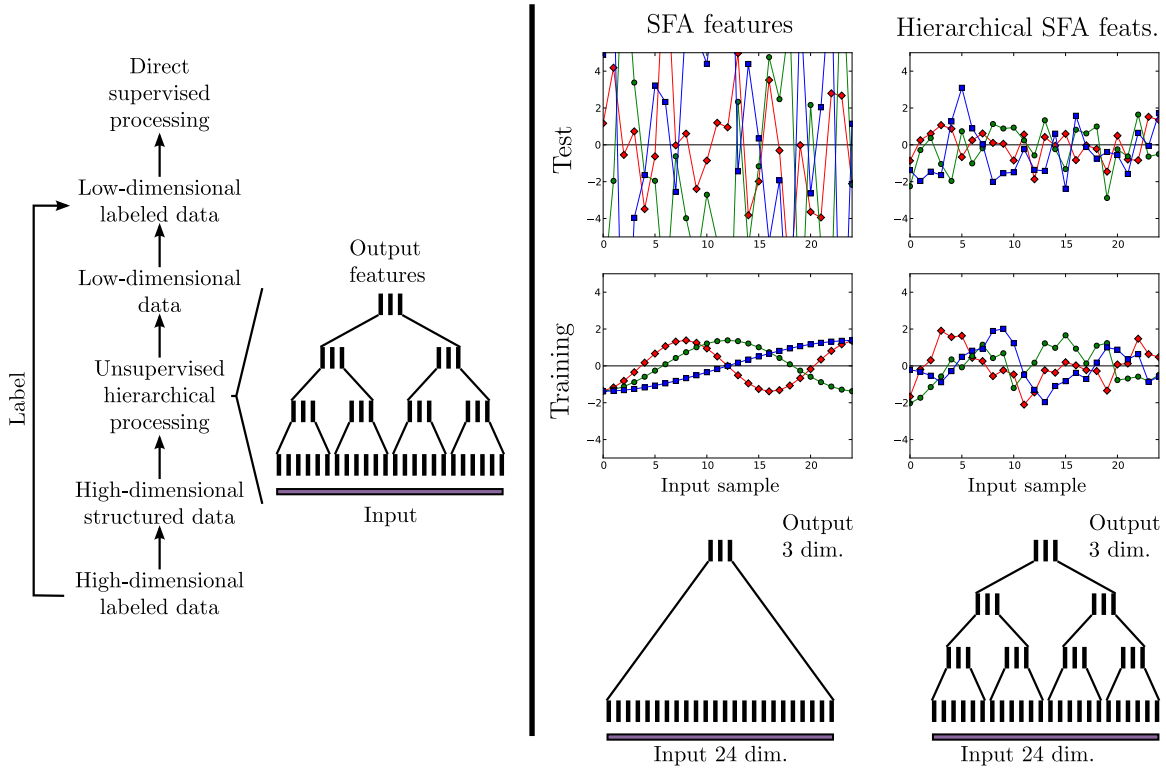


Figure 1: (Left) Transformation of a supervised learning problem on high-dimensional data into a supervised learning problem on low-dimensional data by means of unsupervised hierarchical processing on structured data, that is, without labels. This construction allows the solution of supervised learning problems on high-dimensional data when the dimensionality and number of samples make the direct application of many conventional supervised algorithms infeasible. (Right) Example of how hierarchical SFA (HSFA) is more robust against overfitting than standard SFA. Useless data consisting of 25 random i.i.d. samples is processed by linear SFA and linear HSFA. Both algorithms reduce the dimensionality from 24 to 3 dimensions. Even though the training data is random, the direct application of SFA extracts the slowest features theoretically possible (optimal free responses), which is possible due to the number of dimensions and samples, permitting overfitting. However, it fails to provide consistent features for test data (e.g., standard deviations $\sigma_{\text{training}} = 1.0$ vs. $\sigma_{\text{test}} = 6.5$), indicating lack of generalization. Several points even fall outside the plotted area. In contrast, HSFA extracts much more consistent features (e.g., standard deviations $\sigma_{\text{training}} = 1.0$ vs. $\sigma_{\text{test}} = 1.18$) resulting in less overfitting. Counterintuitively, this result holds even though the HSFA network used has $7 \times 6 \times 3 = 126$ free parameters, many more than the $24 \times 3 = 72$ free parameters of direct SFA.

system, it is a complex computational task. As an example, consider the visual perception of a driver watching pedestrians walking on the street. As the car advances, his receptor responses

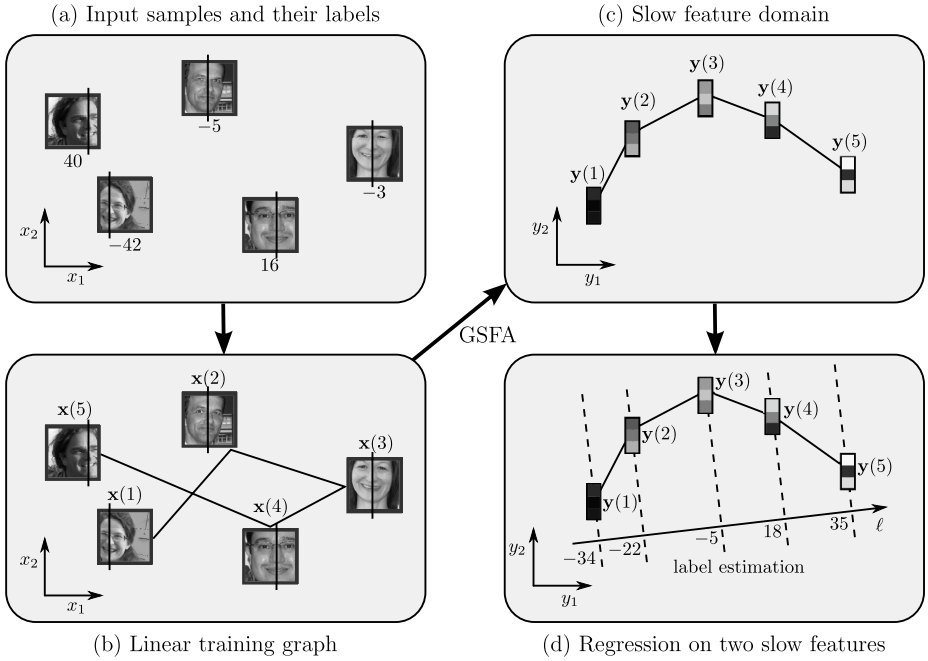


Figure 2: Illustration of the application of GSFA to solve a regression problem. (a) The input samples are 128×128-pixel images with labels indicating the horizontal position of the center of the face. (b) A training graph is constructed using the label information. In this example, only images with most similar labels are connected resulting in a linear graph. (c) The data dimensionality is reduced with GSFA, yielding in this case 3-dimensional feature vectors plotted in the first two dimensions. (d) The application of standard regression methods to the slow features (e.g., linear regression) generates the label estimates. In theory, the labels can be estimated from y_1 alone. In practice, performance is usually improved by using not one, but a few slow features.

typically change quite quickly, and are especially sensitive to the eye movement and to variations in the position or pose of the pedestrians. However, a lot of information, including the position and identity of the pedestrians, can still be distinguished. Relevant abstract information derived from the perception of the environment typically changes on a time scale much slower than the individual sensory inputs. This observation inspires the slowness principle, which explicitly requires the extraction of slow features. This principle has probably first been formulated by Hinton (1989), and online learning rules were developed shortly after by Földiák (1991) and Mitchison (1991). The first closed-form algorithm has been developed by Wiskott and is referred to as Slow feature analysis (SFA, Wiskott, 1998; Wiskott and Sejnowski, 2002). The concise formulation of the SFA optimization problem also permits an extended mathematical treatment so that its properties are well understood analytically (Wiskott, 2003; Franzius et al., 2007; Sprekeler and Wiskott, 2011). SFA has the advantage that it is guaranteed to find an optimal solution within the considered function space. It was initially developed for learning invariances in a model of the primate visual system

(Wiskott and Sejnowski, 2002; Franzius et al., 2011). Berkes and Wiskott (2005) subsequently used it for learning complex-cell receptive fields and Franzius et al. (2007) for place cells in the hippocampus. Recently, researchers have begun using SFA for various technical applications (see Escalante-B. and Wiskott, 2012, for a review).

2.2 Standard SFA Optimization Problem

The SFA optimization problem can be stated as follows (Wiskott, 1998; Wiskott and Sejnowski, 2002; Berkes and Wiskott, 2005). Given an I-dimensional input signal $\mathbf{x}(t) = (x_1(t), \ldots, x_I(t))^T$, where $t \in \mathbb{R}$, find an instantaneous vectorial function $\mathbf{g} : \mathbb{R}^I \to \mathbb{R}^J$ within a function space $\mathcal{F}$, that is, $\mathbf{g}(\mathbf{x}(t)) = (g_1(\mathbf{x}(t)), \ldots, g_J(\mathbf{x}(t)))^T$, such that for each component $y_j(t) \stackrel{\text{def}}{=} g_j(\mathbf{x}(t))$ of the output signal $\mathbf{y}(t) \stackrel{\text{def}}{=} \mathbf{g}(\mathbf{x}(t))$, for $1 \le j \le J$, the objective function

$$\Delta(y_j) \stackrel{\text{def}}{=} \langle \dot{y}_j(t)^2 \rangle_t \text{ is minimal } \textbf{(delta value)} \tag{1}$$

under the constraints

$$\langle y_j(t) \rangle_t = 0 \textbf{ (zero mean),} \tag{2}$$

$$\langle y_j(t)^2 \rangle_t = 1 \textbf{ (unit variance),} \tag{3}$$

$$\langle y_j(t) y_{j'}(t) \rangle_t = 0, \forall j' < j \textbf{ (decorrelation and order)}. \tag{4}$$

The delta value $\Delta(y_j)$ is defined as the time average ($\langle \cdot \rangle_t$) of the squared derivative of y_j and is therefore a measure of the slowness (or rather fastness) of the signal. The constraints (2–4) assure that the output signals are normalized, not constant, and represent different features of the input signal. The problem can be solved iteratively beginning with y_1 (the slowest feature extracted) and finishing with y_J (an algorithm is described in the next section). Due to constraint (4), the delta values are ordered, that is, $\Delta(y_1) \le \Delta(y_2) \le \cdots \le \Delta(y_J)$. See Figure 3 for an illustrative example.

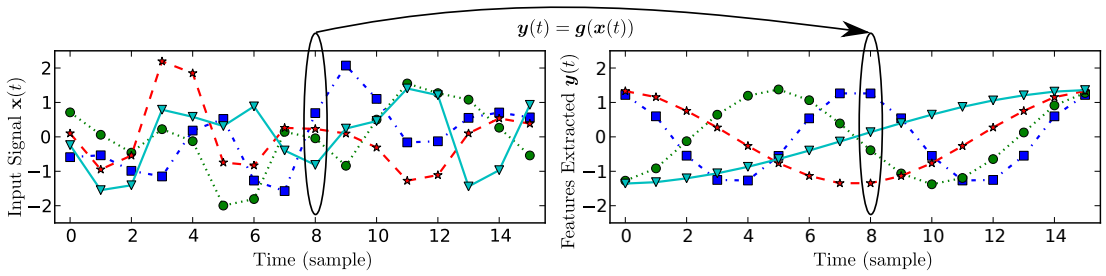


Figure 3: Illustrative example of feature extraction from a 10-dimensional (discrete time) input signal. Four arbitrary components of the input (left) and the four slowest outputs (right) are shown. Notice that feature extraction is an instantaneous operation, even though the outputs are slow over time. This example was designed such that the features extracted are the slowest ones theoretically possible.

In practice, the function $\mathbf{g}$ is usually restricted to a finite-dimensional space $\mathcal{F}$, for example, to all quadratic or linear functions. Highly complex function spaces $\mathcal{F}$ should be avoided because

they result in overfitting. In extreme cases one obtains features such as those in Figure 3 (right) even when the hidden parameters of the input data lack such a precise structure. The problem is then evident when one extracts unstructured features from test data, see Figure 1 (right). An unrestricted function space is, however, useful for various theoretical analyses (e.g., Wiskott, 2003) because of its generality and mathematical convenience.

2.3 Standard Linear SFA Algorithm

The SFA algorithm is typically nonlinear. Even though kernelized versions have been proposed (Bray and Martinez, 2003; Vollgraf and Obermayer, 2006; Böhmer et al., 2012), it is usually implemented more directly with a nonlinear expansion of the input data followed by linear SFA in the expanded space. In this section, we recall the standard linear SFA algorithm (Wiskott and Sejnowski, 2002), in which $\mathcal{F}$ is the space of all linear functions. Discrete time, $t \in \mathbb{N}$, is used for the application of the algorithm to real data. Also the objective function and the constraints are adapted to discrete time. The input is then a single training signal (i.e., a sequence of N samples) $\mathbf{x}(t)$, where $1 \leq t \leq N$, and the time derivative of $\mathbf{x}(t)$ is usually approximated by a sequence of differences of consecutive samples: $\dot{\mathbf{x}}(t) \stackrel{\text{def}}{\approx} \mathbf{x}(t+1) - \mathbf{x}(t)$, for $1 \leq t \leq N-1$.

The output components take the form $g_j(\mathbf{x}) = \mathbf{w}_j^T(\mathbf{x} - \bar{\mathbf{x}})$, where $\bar{\mathbf{x}} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{t=1}^{N}\mathbf{x}(t)$ is the average sample, which is subtracted, so that the output has zero-mean to conform with (2). Thus, in the linear case, the SFA problem reduces to finding an optimal set of weight vectors $\{\mathbf{w}_j\}$ under the constraints above, and it can be solved by linear algebra methods, see below.

The covariance matrix is approximated by the sample covariance matrix

$$\mathbf{C} = \frac{1}{N-1}\sum_{t=1}^{N}(\mathbf{x}(t) - \bar{\mathbf{x}})(\mathbf{x}(t) - \bar{\mathbf{x}})^T ,$$

and the derivative second-moment matrix $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ is approximated as

$$\dot{\mathbf{C}} = \frac{1}{N-1}\sum_{t=1}^{N-1}(\mathbf{x}(t+1) - \mathbf{x}(t))(\mathbf{x}(t+1) - \mathbf{x}(t))^T .$$

Then, a sphered signal $\mathbf{z} \stackrel{\text{def}}{=} \mathbf{S}^T\mathbf{x}$ is computed, such that $\mathbf{S}^T\mathbf{C}\mathbf{S} = \mathbf{I}$ for a sphering matrix $\mathbf{S}$. Afterwards, the J directions of least variance in the derivative signal $\dot{\mathbf{z}}$ are found and represented by an $I \times J$ rotation matrix $\mathbf{R}$, such that $\mathbf{R}^T\dot{\mathbf{C}}_z\mathbf{R} = \mathbf{\Lambda}$, where $\dot{\mathbf{C}}_z \stackrel{\text{def}}{=} \langle \dot{\mathbf{z}}\dot{\mathbf{z}}^T \rangle_t$ and $\mathbf{\Lambda}$ is a diagonal matrix with diagonal elements $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_J$. Finally the algorithm returns the weight matrix $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_J)$, defined as $\mathbf{W} = \mathbf{S}\mathbf{R}$, the features extracted $\mathbf{y} = \mathbf{W}^T(\mathbf{x} - \bar{\mathbf{x}})$, and $\Delta(y_j) = \lambda_j$, for $1 \leq j \leq J$. The linear SFA algorithm is guaranteed to find an optimal solution to the optimization problem (1–4) in the linear function space, for example, the first component extracted is the slowest possible linear feature. A more detailed description of the linear SFA algorithm is provided by Wiskott and Sejnowski (2002).

The complexity of the linear SFA algorithm described above is $O(NI^2 + I^3)$ where N is the number of samples and I is the input dimensionality (possibly after a nonlinear expansion), thus for high-dimensional data standard SFA is not feasible.[1] In practice, it has a speed comparable to PCA, even though SFA also takes into account the temporal structure of the data.

1. The problem is still feasible if N is small enough so that one might apply singular value decomposition methods. However, a small number of samples $N < I$ usually results in pronounced overfitting.

2.4 Hierarchical SFA

To reduce the complexity of SFA, a divide-and-conquer strategy to extract slow features is usually effective (e.g., Franzius et al., 2011). For instance, one can spatially divide the data into lower-dimensional blocks of dimension $I' \ll I$ and extract $J' < I'$ local slow features separately with different instances of SFA, the so-called SFA nodes. Then, one uses another SFA node in a next layer to extract global slow features from the local slow features. Since each SFA node performs dimensionality reduction, the input dimension of the top SFA node is much less than I. This strategy can be repeated iteratively until the input dimensionality at each node is small enough, resulting in a multi-layer hierarchical network. Due to information loss before the top node, this does not guarantee optimal global slow features anymore. However it has shown to be effective in many practical experiments, in part because low-level features are spatially localized in most real data.

Interestingly, hierarchical processing can also be seen as a regularization method, as shown in Figure 1 (right), leading to better generalization. An additional advantage is that the nonlinearity accumulates across layers, so that even when using simple expansions the network as a whole can realize a complex nonlinearity (Escalante-B. and Wiskott, 2011).

3. Graph-Based SFA (GSFA)

In this section, we first present a generalized representation of the training data used by SFA called training graph. Afterwards, we propose the GSFA optimization problem, which is defined in terms of the nodes, edges and weights of such a graph. Then, we present the GSFA algorithm and a probabilistic model for the generation of training data, connecting SFA and GSFA.

3.1 Organization of the Training Samples in a Graph

Learning from a single (multi-dimensional) time series (i.e., a sequence of samples), as in standard SFA, is motivated from biology, because the input data is assumed to originate from sensory perception. In a more technical and supervised learning setting, the training data need not be a time series but may be a set of independent samples. However, one can use the labels to induce structure. For instance, face images may come from different persons and different sources but can still be ordered by, say, age. If one arranges these images in a sequence of increasing age, they would form a linear structure that could be used for training much like a regular time series.

The central contribution of this work is the consideration of a more complex structure for training SFA called training graph. In the example above, one can then introduce a weighted edge between any pair of face images according to some similarity measure based on age (or other criteria such as gender, race, or mimic expression), with high similarity resulting in large edge weights. The original SFA objective then needs to be adapted such that samples connected by large edge weights yield similar output values.

In mathematical terms, the training data is represented as a training graph $G = (\mathbf{V}, \mathbf{E})$ (illustrated in Figure 4) with a set $\mathbf{V}$ of vertices $\mathbf{x}(n)$ (each vertex/node being a sample), and a set $\mathbf{E}$ of edges $(\mathbf{x}(n), \mathbf{x}(n'))$, which are pairs of samples, with $1 \leq n, n' \leq N$. The index n (or n') substitutes the time variable t. The edges are undirected and have symmetric weights

$$\gamma_{n,n'} = \gamma_{n',n} \tag{5}$$

that indicate the similarity between the connected vertices; also each vertex $\mathbf{x}(n)$ has an associated weight $v_n > 0$, which can be used to reflect its importance, frequency, or reliability. For instance, a sample occurring frequently in an observed phenomenon should have a larger weight than a rare sample. This representation includes the standard time series as a special case in which the graph has a linear structure and all node and edge weights are identical (Figure 4.b). How exactly edge weights are derived from label values will be elaborated later.

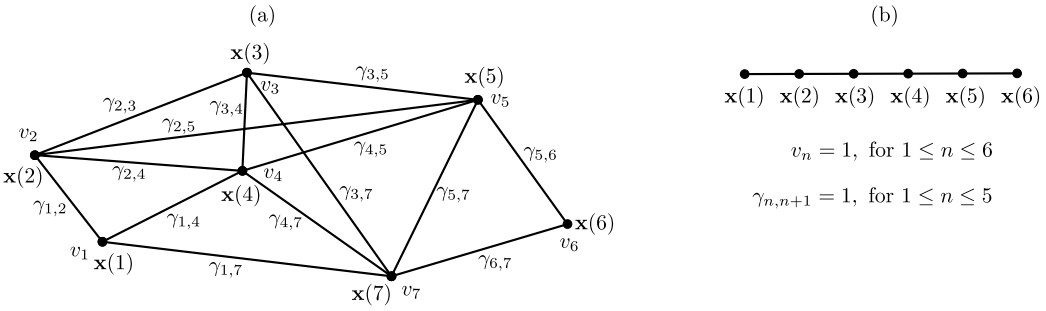


Figure 4: (a) Example of a training graph with $N = 7$ vertices. (b) A regular sample sequence (time-series) represented as a linear graph suitable for GSFA.

3.2 GSFA Optimization Problem

We extend the concept of slowness, originally conceived for sequences of samples, to data structured in a training graph making use of its associated edge weights. The generalized optimization problem can then be formalized as follows. For $1 \le j \le J$, find features $y_j(n)$, where $1 \le n \le N$, such that the objective function

$$\Delta_j \stackrel{\text{def}}{=} \frac{1}{R} \sum_{n,n'} \gamma_{n,n'} (y_j(n') - y_j(n))^2 \text{ is minimal } \textbf{(weighted delta value)} \tag{6}$$

under the constraints

$$\frac{1}{Q} \sum_n v_n y_j(n) = 0 \textbf{ (weighted zero mean)}, \tag{7}$$

$$\frac{1}{Q} \sum_n v_n (y_j(n))^2 = 1 \textbf{ (weighted unit variance)}, \text{ and} \tag{8}$$

$$\frac{1}{Q} \sum_n v_n y_j(n) y_{j'}(n) = 0, \text{ for } j' < j \textbf{ (weighted decorrelation)}, \tag{9}$$

with

$$R \stackrel{\text{def}}{=} \sum_{n,n'} \gamma_{n,n'}, \tag{10}$$

$$Q \stackrel{\text{def}}{=} \sum_n v_n. \tag{11}$$

Compared to the original SFA problem, the vertex weights generalize the normalization constraints, whereas the edge weights extend the objective function to penalize the difference between the outputs of arbitrary pairs of samples. Of course, the factor $1/R$ in the objective function is not essential for the minimization problem. Likewise, the factor $1/Q$ can be dropped from (7–9). These factors, however, provide invariance to the scale of the edge weights as well as to the scale of the node weights, and serve a normalization purpose.

By definition (see Section 3.1), training graphs are undirected and have symmetric edge weights. This does not cause any loss of generality and is justified by the GSFA optimization problem above. Its objective function (6) is insensitive to the direction of an edge because the sign of the output difference cancels out during the computation of Δ_j. It therefore makes no difference whether we choose $\gamma_{n,n'} = 2$ and $\gamma_{n',n} = 0$ or $\gamma_{n,n'} = \gamma_{n',n} = 1$, for instance. We note also that $\gamma_{n,n}$ multiplies with zero in (6) and only enters into the calculation of R. The variables $\gamma_{n,n}$ are kept only for mathematical convenience.

3.3 Linear Graph-Based SFA Algorithm (Linear GSFA)

Similarly to the standard linear SFA algorithm, which solves the standard SFA problem in the linear function space, here we propose an extension that computes an optimal solution to the GSFA problem within the same space. Let the vertices $\mathbf{V} = \{\mathbf{x}(1), \ldots, \mathbf{x}(N)\}$ be the input samples with weights $\{v_1, \ldots, v_N\}$ and the edges $\mathbf{E}$ be the set of edges $(\mathbf{x}(n), \mathbf{x}(n'))$ with edge weights $\gamma_{n,n'}$. To simplify notation we introduce zero edge weights $\gamma_{n,n'} = 0$ for non-existing edges $(\mathbf{x}(n), \mathbf{x}(n')) \notin \mathbf{E}$. The linear GSFA algorithm differs from the standard version only in the computation of the matrices $\mathbf{C}$ and $\dot{\mathbf{C}}$, which now take into account the neighbourhood structure (samples, edges, and weights) specified by the training graph.

The sample covariance matrix $\mathbf{C_G}$ is defined as:

$$\mathbf{C_G} \stackrel{\text{def}}{=} \frac{1}{Q}\sum_n v_n(\mathbf{x}(n)-\hat{\mathbf{x}})(\mathbf{x}(n)-\hat{\mathbf{x}})^T = \frac{1}{Q}\sum_n \left(v_n\mathbf{x}(n)(\mathbf{x}(n))^T\right) - \hat{\mathbf{x}}\hat{\mathbf{x}}^T, \tag{12}$$

where

$$\hat{\mathbf{x}} \stackrel{\text{def}}{=} \frac{1}{Q}\sum_n v_n\mathbf{x}(n) \tag{13}$$

is the weighted average of all samples. The derivative second-moment matrix $\dot{\mathbf{C}}_\mathbf{G}$ is defined as:

$$\dot{\mathbf{C}}_\mathbf{G} \stackrel{\text{def}}{=} \frac{1}{R}\sum_{n,n'} \gamma_{n,n'}\left(\mathbf{x}(n')-\mathbf{x}(n)\right)\left(\mathbf{x}(n')-\mathbf{x}(n)\right)^T. \tag{14}$$

Given these matrices, the computation of $\mathbf{W}$ is the same as in the standard algorithm (Section 2.3). Thus, a sphering matrix $\mathbf{S}$ and a rotation matrix $\mathbf{R}$ are computed with

$$\mathbf{S}^T\mathbf{C_G}\mathbf{S} = \mathbf{I}, \text{ and} \tag{15}$$

$$\mathbf{R}^T\mathbf{S}^T\dot{\mathbf{C}}_\mathbf{G}\mathbf{S}\mathbf{R} = \mathbf{\Lambda}, \tag{16}$$

where $\mathbf{\Lambda}$ is a diagonal matrix with diagonal elements $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_J$. Finally the algorithm returns $\Delta(y_1), \ldots, \Delta(y_J)$, $\mathbf{W}$ and $\mathbf{y}(n)$, where

$$\mathbf{W} = \mathbf{S}\mathbf{R}, \text{ and} \tag{17}$$

$$\mathbf{y}(n) = \mathbf{W}^T(\mathbf{x}(n)-\hat{\mathbf{x}}). \tag{18}$$

3.4 Correctness of the Graph-Based SFA Algorithm

We now prove that the GSFA algorithm indeed solves the optimization problem (6–9). This proof is similar to the optimality proof of the standard SFA algorithm (Wiskott and Sejnowski, 2002). For simplicity, assume that $\mathbf{C_G}$ and $\dot{\mathbf{C}}_\mathbf{G}$ have full rank.

The weighted zero mean constraint (7) holds trivially for any $\mathbf{W}$, because

$$\begin{aligned}\sum_n v_n \mathbf{y}(n) &\overset{(18)}{=} \sum_n v_n \mathbf{W}^T (\mathbf{x}(n) - \hat{\mathbf{x}}) \\ &= \mathbf{W}^T \left(\sum_n v_n \mathbf{x}(n) - \sum_{n'} v_{n'} \hat{\mathbf{x}} \right) \\ &\overset{(13,11)}{=} \mathbf{W}^T (Q\hat{\mathbf{x}} - Q\hat{\mathbf{x}}) = \mathbf{0}.\end{aligned}$$

We also find

$$\begin{aligned}\mathbf{I} &= \mathbf{R}^T \mathbf{I} \mathbf{R} \quad \text{(since } \mathbf{R} \text{ is a rotation matrix)}, \\ &\overset{(15)}{=} \mathbf{R}^T (\mathbf{S}^T \mathbf{C_G} \mathbf{S}) \mathbf{R}, \\ &\overset{(17)}{=} \mathbf{W}^T \mathbf{C_G} \mathbf{W}, \\ &\overset{(12)}{=} \mathbf{W}^T \frac{1}{Q} \sum_n v_n (\mathbf{x}(n) - \hat{\mathbf{x}})(\mathbf{x}(n) - \hat{\mathbf{x}})^T \mathbf{W}, \\ &\overset{(18)}{=} \frac{1}{Q} \sum_n v_n \mathbf{y}(n) (\mathbf{y}(n))^T,\end{aligned}$$

which is equivalent to the normalization constraints (8) and (9).

Now, let us consider the objective function

$$\begin{aligned}\Delta_j &\overset{(6)}{=} \frac{1}{R} \sum_{n,n'} \gamma_{n,n'} \big(y_j(n') - y_j(n)\big)^2 \\ &\overset{(14)}{=} \mathbf{w}_j^T \dot{\mathbf{C}}_\mathbf{G} \mathbf{w}_j \\ &\overset{(17)}{=} \mathbf{r}_j^T \mathbf{S}^T \dot{\mathbf{C}}_\mathbf{G} \mathbf{S} \mathbf{r}_j \\ &\overset{(16)}{=} \lambda_j,\end{aligned}$$

where $\mathbf{R} = (\mathbf{r}_1, \ldots, \mathbf{r}_J)$. The algorithm finds a rotation matrix $\mathbf{R}$ solving (16) and yielding increasing λs. It can be seen (cf. Adali and Haykin, 2010, Section 4.2.3) that this $\mathbf{R}$ also achieves the minimization of Δ_j, for $j = 1, \ldots, J$, hence, fulfilling (6).

3.5 Probabilistic Interpretation of Training Graphs

In this section, we give an intuition for the relationship between GSFA and standard SFA. Readers less interested in this theoretical excursion can safely skip it. This section is inspired in part by the Markov chain introduced by Klampfl and Maass (2010).

Given a training graph, we construct below a Markov chain $\mathcal{M}$ for the generation of input data such that training standard SFA with such data yields the same features as GSFA does with the

graph. Contrary to the graph introduced by Klampfl and Maass (2010), the formulation here is not restricted to classification, accounting for any training graph irrespective of its purpose, and there is one state per sample rather than one state per class. In order for the equivalence of GSFA and SFA to hold, the vertex weights $\tilde{v}_n$ and edge weights $\tilde{\gamma}_{n,n'}$ of the graph must fulfil the following *normalization restrictions*:

$$\sum_n \tilde{v}_n \;=\; 1\,, \tag{19}$$

$$\sum_{n'} \tilde{\gamma}_{n,n'}/\tilde{v}_n \;=\; 1 \quad \forall n\,, \tag{20}$$

$$\overset{(5)}{\Longleftrightarrow} \quad \sum_{n'} \tilde{\gamma}_{n',n}/\tilde{v}_n \;=\; 1 \quad \forall n\,, \tag{21}$$

$$\sum_{n,n'} \tilde{\gamma}_{n,n'} \overset{(20,19)}{=} 1\,. \tag{22}$$

Restrictions (19) and (22) can always be assumed without loss of generality, because they can be achieved by a constant scaling of the weights (i.e., $\tilde{v}_n \leftarrow \tilde{v}_n/Q$, $\tilde{\gamma}_{n,n'} \leftarrow \tilde{\gamma}_{n,n'}/R$) without affecting the outputs generated by GSFA. Restriction (20) is fundamental because it limits the graph connectivity, and indicates (after multiplying with $\tilde{v}_n$) that each vertex weight should be equal to the sum of the weights of all edges originating from such a vertex.

The Markov chain is then a sequence $\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3, \ldots$ of random variables that can assume states that correspond to different input samples. $\mathbf{Z}_1$ is drawn from the initial distribution $\boldsymbol{p}_1$, which is equal to the stationary distribution $\boldsymbol{\pi}$, where

$$\pi_n = \boldsymbol{p}_1(n) \overset{\text{def}}{=} \Pr(\mathbf{Z}_1 = \mathbf{x}(n)) \overset{\text{def}}{=} \tilde{v}_n\,, \tag{23}$$

and the transition probabilities are given by

$$P_{nn'} \overset{\text{def}}{=} \Pr(\mathbf{Z}_{t+1} = \mathbf{x}(n') | \mathbf{Z}_t = \mathbf{x}(n)) \;\overset{\text{def}}{=}\; (1-\varepsilon)\tilde{\gamma}_{n,n'}/\tilde{v}_n + \varepsilon\tilde{v}_{n'} \underset{\lim_{\varepsilon\to 0}}{=} \tilde{\gamma}_{n,n'}/\tilde{v}_n\,, \tag{24}$$

$$\overset{(23)}{\Longrightarrow} \quad \Pr(\mathbf{Z}_{t+1} = \mathbf{x}(n'), \mathbf{Z}_t = \mathbf{x}(n)) \;=\; (1-\varepsilon)\tilde{\gamma}_{n,n'} + \varepsilon\tilde{v}_n\tilde{v}_{n'} \underset{\lim_{\varepsilon\to 0}}{=} \tilde{\gamma}_{n,n'}\,, \tag{25}$$

(for $\mathbf{Z}_t$ stationary) with $0 < \varepsilon \ll 1$. Due to the ε-term all states of the Markov chain can transition to all other states including themselves, which makes the Markov chain irreducible and aperiodic, and therefore ergodic. Thus, the stationary distribution is unique and the Markov chain converges to it. The normalization restrictions (19), (20), and (22) ensure the normalization of (23), (24), and (25), respectively.

It is easy to see that $\boldsymbol{\pi} = \{\tilde{v}_n\}_{n=1}^N$ is indeed a stationary distribution, since for $\mathbf{p}_t(n) = \tilde{v}_n$

$$\begin{aligned}
\mathbf{p}_{t+1}(n) \;=\; \Pr(\mathbf{Z}_{t+1} = \mathbf{x}(n)) \;&=\; \sum_{n'} \Pr(\mathbf{Z}_{t+1} = \mathbf{x}(n) | \mathbf{Z}_t = \mathbf{x}(n')) \Pr(\mathbf{Z}_t = \mathbf{x}(n')) \\
&\overset{(23,24)}{=} \sum_{n'} \left((1-\varepsilon)\left(\tilde{\gamma}_{n',n}/\tilde{v}_{n'}\right) + \varepsilon\tilde{v}_n \right) \tilde{v}_{n'} \\
&\overset{(21,19)}{=} (1-\varepsilon)\tilde{v}_n + \varepsilon\tilde{v}_n \;=\; \tilde{v}_n \;=\; \mathbf{p}_t(n)\,.
\end{aligned} \tag{26}$$

The time average of the input sequence is

$$\begin{aligned}\boldsymbol{\mu}_Z &\stackrel{\text{def}}{=} \langle \mathbf{Z}_t \rangle_t \\ &= \langle \mathbf{Z} \rangle_{\boldsymbol{\pi}} \quad \text{(since } \mathcal{M} \text{ is ergodic)} \\ &\stackrel{(26)}{=} \sum_n \tilde{v}_n \mathbf{x}(n) \\ &\stackrel{(13)}{=} \hat{\mathbf{x}}, \end{aligned} \tag{27}$$

and the covariance matrix is

$$\begin{aligned}\mathbf{C} &\stackrel{\text{def}}{=} \langle (\mathbf{Z}_t - \boldsymbol{\mu}_Z)(\mathbf{Z}_t - \boldsymbol{\mu}_Z)^T \rangle_t \\ &\stackrel{(27)}{=} \langle (\mathbf{Z} - \hat{\mathbf{x}})(\mathbf{Z} - \hat{\mathbf{x}})^T \rangle_{\boldsymbol{\pi}} \quad \text{(since } \mathcal{M} \text{ is ergodic)} \\ &\stackrel{(26)}{=} \sum_n \tilde{v}_n (\mathbf{x}(n) - \hat{\mathbf{x}})(\mathbf{x}(n) - \hat{\mathbf{x}})^T \\ &\stackrel{(12)}{=} \mathbf{C}_{\mathbf{G}}, \end{aligned}$$

whereas the derivative covariance matrix is

$$\begin{aligned}\dot{\mathbf{C}} &\stackrel{\text{def}}{=} \langle \dot{\mathbf{Z}}_t \dot{\mathbf{Z}}_t^T \rangle_t \\ &= \langle \dot{\mathbf{Z}} \dot{\mathbf{Z}}^T \rangle_{\boldsymbol{\pi}} \quad \text{(since } \mathcal{M} \text{ is ergodic)} \\ &\stackrel{(25)}{=} \sum_{n,n'} \left((1-\varepsilon)\tilde{\gamma}_{n,n'} + \varepsilon \tilde{v}_n \tilde{v}_{n'} \right) (\mathbf{x}(n') - \mathbf{x}(n))(\mathbf{x}(n') - \mathbf{x}(n))^T, \end{aligned} \tag{28}$$

where $\dot{\mathbf{Z}}_t \stackrel{\text{def}}{=} \mathbf{Z}_{t+1} - \mathbf{Z}_t$. Notice that $\lim_{\varepsilon \to 0} \dot{\mathbf{C}} \stackrel{(28)}{=} \tilde{\gamma}_{n,n'} (\mathbf{x}(n') - \mathbf{x}(n))(\mathbf{x}(n') - \mathbf{x}(n))^T \stackrel{(14)}{=} \dot{\mathbf{C}}_{\mathbf{G}}$. Therefore, if a graph fulfils the normalization restrictions (19)–(22), GSFA yields the same features as standard SFA on the sequence generated by the Markov chain, in the limit $\varepsilon \to 0$.

3.6 Construction of Training Graphs

One can, in principle, construct training graphs with arbitrary connections and weights. However, when the goal is to solve a supervised learning task, the graph created should implicitly integrate the label information. An appropriate structure of the training graphs depends on whether the goal is classification or regression. In the next sections, we describe each case separately. We have previously implemented the proposed training graphs, and we have tested and verified their usefulness on real-world data (Escalante-B. and Wiskott, 2010; Mohamed and Mahdi, 2010; Stallkamp et al., 2011; Escalante-B. and Wiskott, 2012).

4. Classification with SFA

In this section, we show how to use GSFA to profit from the label information and solve classification tasks more efficiently and accurately than with standard SFA.

4.1 Clustered Training Graph

To generate features useful for classification, we propose the use of a *clustered training graph* presented below (Figure 5). Assume there are S identities/classes, and for each particular identity $s = 1,\ldots,S$ there are N_s samples $\mathbf{x}^s(n)$, where $n = 1,\ldots,N_s$, making a total of $N = \sum_s N_s$ samples. We define the clustered training graph as a graph $G = (\mathbf{V},\mathbf{E})$ with vertices $\mathbf{V} = \{\mathbf{x}^s(n)\}$, and edges $\mathbf{E} = \{(\mathbf{x}^s(n),\mathbf{x}^s(n'))\}$ for $s = 1,\ldots,S$, and $n,n' = 1,\ldots,N_s$. Thus all pairs of samples of the same identity are connected, while samples of different identity are not connected. Node weights are identical and equal to one, that is, $\forall s,n:\ v_n^s = 1$. In contrast, edge weights, $\gamma^s_{n,n'} = 1/N_s \quad \forall n,n'$, depend on the cluster size.[2] Otherwise identities with a large N_s would be over-represented because the number of edges in the complete subgraph for identity s grows quadratically with N_s. These weights directly fulfil the normalization restriction (20). As usual, a trivial scaling of the node and edge weights suffices to fulfil restrictions (19) and (22), allowing the probabilistic interpretation of the graph. The optimization problem associated to this graph explicitly demands that samples from the same object identity should be typically mapped to similar outputs.

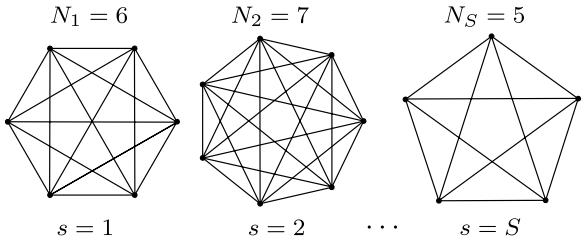


Figure 5: Illustration of a *clustered* training graph used for a classification problem. All samples belonging to the same object identity form fully connected subgraphs. Thus, for S identities there are S complete subgraphs. Self-loops not shown.

4.2 Efficient Learning Using the Clustered Training Graph

At first sight, the large number of edges, $\sum_s N_s(N_s+1)/2$, seems to introduce a computational burden. Here we show that this is not the case if one exploits the symmetry of the clustered training graph. From (12), the sample covariance matrix of this graph using the node weights $v_n^s = 1$ is (notice the definition of Π^s and $\hat{\mathbf{x}}^s$):

$$\mathbf{C}_{\text{clus}} \overset{(12)}{=} \frac{1}{Q}\left(\underbrace{\sum_s \sum_{n=1}^{N_s} \mathbf{x}^s(n)(\mathbf{x}^s(n))^T}_{\overset{\text{def}}{=}\Pi^s} - Q\Big(\underbrace{\frac{1}{Q}\sum_s \overbrace{\sum_{n=1}^{N_s} \mathbf{x}^s(n)}^{\overset{\text{def}}{=}N_s\hat{\mathbf{x}}^s}}_{\overset{(13)}{=}\hat{\mathbf{x}}} \Big)\Big(\frac{1}{Q}\sum_s \sum_{n=1}^{N_s} \mathbf{x}^s(n) \Big)^T \right), \tag{29}$$

$$= \frac{1}{Q}\Big(\sum_s \Pi^s - Q\hat{\mathbf{x}}(\hat{\mathbf{x}})^T \Big), \tag{30}$$

2. These node and edge weights assume that the classification of all samples is equally important. In the alternative case that classification over every cluster is equally important, one can set $v_n^s = 1/N_s$ and $\forall n,n':\ \gamma^s_{n,n'} = (1/N_s)^2$ instead.

where $Q \overset{(11)}{=} \sum_s \sum_{n=1}^{N_s} 1 = \sum_s N_s = N$.

From (14), the derivative covariance matrix of the clustered training graph using edge weights $\gamma^s_{n,n'} = 1/N_s$ is:

$$\dot{\mathbf{C}}_{\text{clus}} \overset{(14)}{=} \frac{1}{R} \sum_s \frac{1}{N_s} \sum_{n,n'=1}^{N_s} (\mathbf{x}^s(n') - \mathbf{x}^s(n))(\mathbf{x}^s(n') - \mathbf{x}^s(n))^T , \tag{31}$$

$$= \frac{1}{R} \sum_s \frac{1}{N_s} \sum_{n,n'=1}^{N_s} \left(\mathbf{x}^s(n')(\mathbf{x}^s(n'))^T + \mathbf{x}^s(n)(\mathbf{x}^s(n))^T - \mathbf{x}^s(n')(\mathbf{x}^s(n))^T - \mathbf{x}^s(n)(\mathbf{x}^s(n'))^T \right),$$

$$\overset{(29)}{=} \frac{1}{R} \sum_s \frac{1}{N_s} \left(N_s \sum_{n=1}^{N_s} \mathbf{x}^s(n)(\mathbf{x}^s(n))^T + N_s \sum_{n'=1}^{N_s} \mathbf{x}^s(n')(\mathbf{x}^s(n'))^T - 2N_s \hat{\mathbf{x}}^s (N_s \hat{\mathbf{x}}^s)^T \right),$$

$$\overset{(29)}{=} \frac{2}{R} \sum_s \left(\Pi^s - N_s \hat{\mathbf{x}}^s (\hat{\mathbf{x}}^s)^T \right), \tag{32}$$

where $R \overset{(10)}{=} \sum_s \sum_{n,n'} \gamma^s_{n,n'} = \sum_s \sum_{n,n'} 1/N_s = \sum_s (N_s)^2/N_s = \sum_s N_s = N$.

The complexity of computing $\mathbf{C}_{\text{clus}}$ using (29) or (30) is the same, namely $O(\sum_s N_s)$ (vector) operations. However, the complexity of computing $\dot{\mathbf{C}}_{\text{clus}}$ can be reduced from $O(\sum_s N_s^2)$ operations directly using (31) to $O(\sum_s N_s)$ operations using (32). This algebraic simplification allows us to compute $\dot{\mathbf{C}}_{\text{clus}}$ with a complexity linear in N (and N_s), which constitutes an important speedup since, depending on the application, N_s might be larger than 100 and sometimes even $N_s > 1000$.

Interestingly, one can show that the features learned by GSFA on this graph are equivalent to those learned by FDA (see Section 7).

4.3 Supervised Step for Classification Problems

Consistent with FDA, the theory of SFA using an unrestricted function space (optimal free responses) predicts that, for this type of problem, the first $S-1$ slow features extracted are orthogonal step functions, and are piece-wise constant for samples from the same identity (Berkes, 2005a). This closely approximates what has been observed empirically, which can be informally described as features that are approximately constant for samples of the same identity, with moderate noise.

When the features extracted are close to the theoretical predictions (e.g., their Δ-values are small), their structure is simple enough that one can use even a modest supervised step after SFA, such as a nearest centroid or a Gaussian classifier (in which a Gaussian distribution is fitted to each class) on $S-1$ slow features or less. We suggest the use of a Gaussian classifier because in practice we have obtained better robustness when enough training data is available. While a more powerful classification method, such as an SVM, might also be used, we have found only a small increase in performance at the cost of longer training times.

5. Regression with SFA

The objective in regression problems is to learn a mapping from samples to labels providing the best estimation as measured by a loss function, for example, the root mean squared error (RMSE) between the estimated labels, $\hat{\ell}$, and their ground-truth values, ℓ. We assume here that the loss function is an increasing function of $|\hat{\ell} - \ell|$ (e.g., contrary to periodic functions useful to compare angular values, or arbitrary functions of $\hat{\ell}$ and ℓ).

Regression problems can be address with SFA through multiple methods. The fundamental idea is to treat labels as the value of a hidden slow parameter that we want to learn. In general, SFA will not extract the label values exactly. However, optimization for slowness implies that samples with similar label values are typically mapped to similar output values. After SFA reduces the dimensionality of the data, a complementary explicit regression step on a few features solves the original regression problem.

In this section, we propose four SFA-based methods that explicitly use available labels. The first method is called *sample reordering* and employs standard SFA, whereas the remaining ones employ GSFA with three different training graphs called *sliding window*, *serial*, and *mixed* (Sections 5.1–5.4). The selection of the explicit regression step for post-processing is discussed in Section 5.5.

5.1 Sample Reordering

Let $\mathbf{X}' = (\mathbf{x}'(1), \ldots, \mathbf{x}'(N))$ be a sequence of N data samples with labels $\boldsymbol{\ell}' = (\ell'_1, \ldots, \ell'_N)$. The data is reordered by means of a permutation $\pi(\cdot)$ in such a way that the labels become monotonically increasing. The reordered samples are $\mathbf{X} = (\mathbf{x}(1), \ldots, \mathbf{x}(N))$, where $\mathbf{x}(n) = \mathbf{x}'(\pi(n))$, and their labels are $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_N)$ with $\ell_l \leq \ell_{l+1}$. Afterwards the sequence $\mathbf{X}$ is used to train standard SFA using the regular single-sequence method (Figure 6).

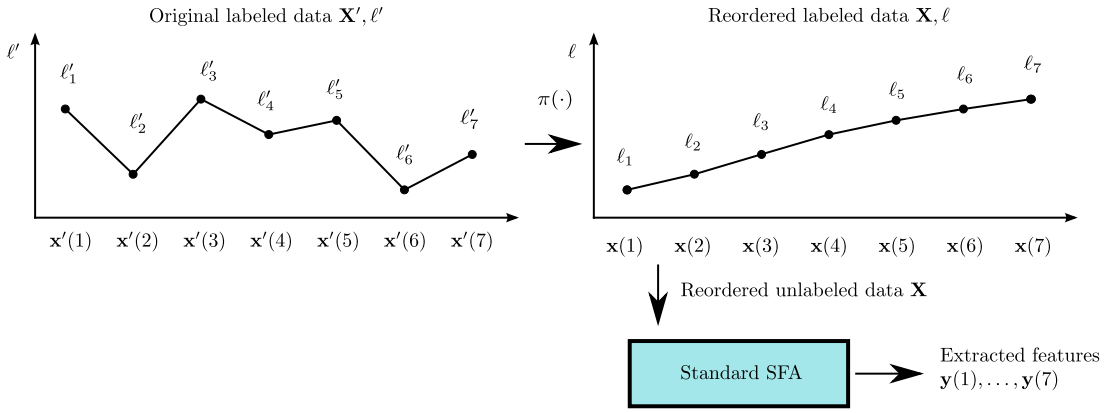


Figure 6: Sample reordering approach. Standard SFA is trained with a reordered sample sequence, in which the hidden labels are increasing.

Since the ordered label values only increase, they change very slowly and should be found by SFA (or actually some increasing/decreasing function of the labels that also fulfils the normalization conditions). Clearly, SFA could only extract this information if the samples indeed intrinsically contain information about the labels such that it is possible to extract the labels from them. Due to limitations of the feature space considered, insufficient data, noise, etc., one typically obtains noisy and distorted versions of the predicted signals.

In this basic approach, the computation of the covariance matrices takes $O(N)$ operations. Since this method only requires standard SFA and is the most straightforward to implement, we recommend its use for first experiments. If more robust outputs are desired, the methods below based on GSFA are more appropriate.

5.2 Sliding Window Training Graph

This is an improvement over the method above in which GSFA facilitates the consideration of more connections. Starting from the reordered sequence $\mathbf{X}$ as defined above, a training graph is constructed, in which each sample $\mathbf{x}(n)$ is connected to its d closest samples to the left and to the right in the order given by $\mathbf{X}$. Thus, $\mathbf{x}(n)$ is connected to the samples $\mathbf{x}(n-d),\ldots,\mathbf{x}(n-1)$, $\mathbf{x}(n+1),\ldots,\mathbf{x}(n+d)$ (Figure 7.a). In this graph, the vertex weights are constant, that is, $v_n = 1$, and the edge weights typically depend on the distance of the samples involved, that is, $\forall n,n' : \gamma_{n,n'} = f(|n'-n|)$, for some function $f(\cdot)$ that specifies the shape of a "weight window". The simplest case is a square weight window defined by $\gamma_{n,n'} = 1$ if $|n'-n| \leq d$ and $\gamma_{n,n'} = 0$ otherwise. For the experiments in this article, we employ a *mirrored* sliding window with edge weights

$$\gamma_{n,n'} = \begin{cases} 2, & \text{if} \quad n+n' \leq d+1 \text{ or } n+n' \geq 2N-1\,, \\ 1, & \text{if} \quad |n'-n| \leq d,\, n+n' > d+1 \text{ and } n+n' < 2N-1\,, \\ 0, & \text{otherwise}\,. \end{cases}$$

These weights compensate the limited connectivity of the few first and last samples (which are connected by d to $2d-1$ edges) in contrast to intermediate samples (connected by $2d$ edges). Preliminary experiments suggest that such compensation slightly improves the quality of the extracted features, as explained below.

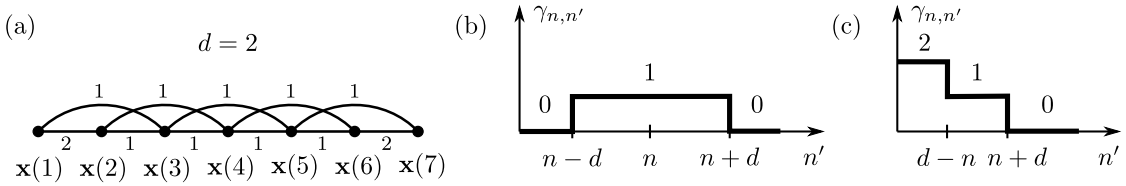


Figure 7: (a) A mirrored square sliding window training graph with a half-width of $d = 2$. Each vertex is thus adjacent to at most 4 other vertices. (b) Illustration of the edge weights of an intermediate node $\mathbf{x}(n)$ for an arbitrary window half-width d. (c) Edge weights for a node $\mathbf{x}(n)$ close to the left extreme ($n < d$). Notice that the sum of the edge weights is also approximately $2d$ for extreme nodes.

GSFA is guaranteed to find functions that minimize (6) within the function space considered. However, whether such a solution is suitable for regression largely depends on how one has defined the weights of the training graph. For instance, if there is a sample with a large node weight that has only weak connections to the other samples, an optimal but undesired solution might assign a high positive value to that single sample and negative small values to all other samples. This can satisfy the zero mean and unit variance constraint while yielding a small Δ-value, because the large differences in output value only occur at the weak connections. Thus, this is a good solution in terms of the optimization problem but not a good one to solve the regression problem at hand, because the samples with small values are hard to discriminate. We refer to such solutions as pathological. Pathological solutions have certain similarities to the features obtained for classification, which are approximately constant for each cluster (class) but discontinuous among them.

The occurrence of pathological solutions depends on the concrete data samples, feature space, and training graph. A necessary condition is that the graph is connected because, as discussed in Section 4, for disconnected graphs GSFA has a strong tendency to produce a representation suitable for classification rather than regression. After various experiments, we have found useful to enforce the normalization restriction (20) at least approximately (after node and edge weights have been normalized). This ensures that the samples are connected sufficiently strongly to the other ones, relative to their own node weight. Of course, one should not resort to self-loops $\gamma_{n,n} \neq 0$ to trivially fulfil the restriction.

The improved continuity of the features appears to also benefit performance after the supervised step. This is the reason why we make the node weights of the first and last groups of samples in the serial training graph weaker, the intra-group connections of the first and last groups of samples in the mixed graph stronger, and introduced mirroring for the square sliding window graph.

In the sliding window training graph with arbitrary window, the computation of $\mathbf{C_G}$ and $\dot{\mathbf{C}}_\mathbf{G}$ requires $O(dN)$ operations. If the window is square (mirrored or not), the computation can be improved to $O(N)$ operations by using accumulators for sums and products and reusing intermediate results. While larger d implies more connections, connecting too distant samples is undesired. The selection of d is non-crucial and done empirically.

5.3 Serial Training Graph

The *serial* training graph is similar to the clustered training graph used for classification in terms of construction and efficiency. It results from discretizing the original labels $\boldsymbol{\ell}$ into a relatively small set of discrete labels of size L, namely $\{\ell_1, \ldots, \ell_L\}$, where $\ell_1 < \ell_2 < \cdots < \ell_L$. As described below, faster training is achieved if L is small, for example, $3 \leq L \ll N$.

In this graph, the vertices are grouped according to their discrete labels. Every sample in the group with label ℓ_l is connected to every sample in the groups with label ℓ_{l+1} and ℓ_{l-1} (except the samples in the first and last groups, which can only be connected to one neighbouring group). The only existing connections are inter-group connections, no intra-group connections are present.

The samples used for training are denoted by $\mathbf{x}^l(n)$, where the index l ($1 \leq l \leq L$) denotes the group (discrete label) and n ($1 \leq n \leq N_l$) denotes the sample within such a group. For simplicity, we assume here that all groups have the same number N_g of samples: $\forall l:\ N_l = N_g$. Thus the total number of samples is $N = LN_g$. The vertex weight of $\mathbf{x}^l(n)$ is denoted by v_n^l, where $v_n^l = 1$ for $l \in \{1, L\}$ and $v_n^l = 2$ for $1 < l < L$. The edge weight of the edge $(\mathbf{x}^l(n), \mathbf{x}^{l+1}(n'))$ is denoted by $\gamma_{n,n'}^{l,l+1}$, and we use the same edge weight for all connections: $\forall n, n', l:\ \gamma_{n,n'}^{l,l+1} = 1$. Thus, all edges have a weight of 1, and all samples are assigned a weight of 2 except for the samples in the first and last groups, which have a weight of 1 (Figure 8). The reason for the different weights in the first and last groups is to improve feature quality by enforcing the normalization restriction (20) (after node and edge weight normalization). Notice that since any two vertices of the same group are adjacent to exactly the same neighbours, they are likely to be mapped to similar outputs by GSFA.

The sum of vertex weights is $Q \overset{(11)}{=} N_g + 2N_g(L-2) + N_g = 2N_g(L-1)$ and the sum of edge weights is $R \overset{(10)}{=} (L-1)(N_g)^2$, which is also the number of connections considered. Unsurprisingly, the structure of the graph can be exploited to train GSFA efficiently. Similarly to the clustered training graph, define the average of the samples from the group l as $\hat{\mathbf{x}}^l \overset{\text{def}}{=} \sum_n \mathbf{x}^l(n)/N_g$, the sum of the products of samples from group l as $\Pi^l = \sum_n x^l(n)(x^l(n))^T$, and the weighted sample average

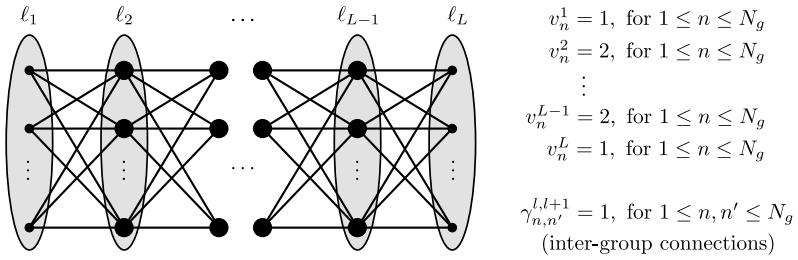


Figure 8: Illustration of a serial training graph with L discrete labels. Even though the original labels of two samples might differ, they will be grouped together if they have the same discrete label. In the figure, a bigger node represents a sample with a larger weight, and the ovals represent the groups.

as:

$$\hat{\mathbf{x}} \stackrel{\text{def}}{=} \frac{1}{Q}\sum_n \left(\mathbf{x}^1(n) + \mathbf{x}^L(n) + 2\sum_{l=2}^{L-1}\mathbf{x}^l(n)\right) = \frac{1}{2(L-1)}\left(\hat{\mathbf{x}}^1 + \hat{\mathbf{x}}^L + 2\sum_{l=2}^{L-1}\hat{\mathbf{x}}^l\right). \tag{33}$$

From (12), the sample covariance matrix accounting for the weights v_n^l of the serial training graph is:

$$\begin{aligned}\mathbf{C}_{\text{ser}} &\stackrel{(12,33)}{=} \frac{1}{Q}\left(\sum_n \mathbf{x}^1(n)(\mathbf{x}^1(n))^T + 2\sum_{l=2}^{L-1}\sum_n \mathbf{x}^l(n)(\mathbf{x}^l(n))^T + \sum_n \mathbf{x}^L(n)(\mathbf{x}^L(n))^T - Q\hat{\mathbf{x}}(\hat{\mathbf{x}})^T\right)\\ &= \frac{1}{Q}\left(\Pi^1 + \Pi^L + 2\sum_{l=2}^{L-1}\Pi^l - Q\hat{\mathbf{x}}'(\hat{\mathbf{x}}')^T\right).\end{aligned}$$

From (14), the matrix $\dot{\mathbf{C}}_{\mathbf{G}}$ using the edges $\gamma_{n,n'}^{l,l+1}$ defined above is:

$$\dot{\mathbf{C}}_{\text{ser}} \stackrel{(14)}{=} \frac{1}{R}\sum_{l=1}^{L-1}\sum_{n,n'}(\mathbf{x}^{l+1}(n') - \mathbf{x}^l(n))(\mathbf{x}^{l+1}(n') - \mathbf{x}^l(n))^T \tag{34}$$

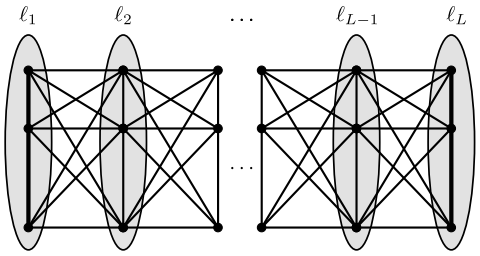
$$\begin{aligned}&= \frac{1}{R}\sum_{l=1}^{L-1}\sum_{n,n'}\left(\mathbf{x}^{l+1}(n')(\mathbf{x}^{l+1}(n'))^T + \mathbf{x}^l(n)(\mathbf{x}^l(n))^T - \mathbf{x}^l(n)(\mathbf{x}^{l+1}(n'))^T - \mathbf{x}^{l+1}(n')(\mathbf{x}^l(n))^T\right)\\ &= \frac{1}{R}\sum_{l=1}^{L-1}\left(\sum_{n'}\left(\Pi^{l+1} + \Pi^l\right) - \left(\sum_n \mathbf{x}^l(n)\right)\left(\sum_{n'}\mathbf{x}^{l+1}(n')\right)^T - \left(\sum_{n'}\mathbf{x}^{l+1}(n')\right)\left(\sum_n \mathbf{x}^l(n)\right)^T\right)\\ &= \frac{N_g}{R}\sum_{l=1}^{L-1}\left(\Pi^{l+1} + \Pi^l - N_g\hat{\mathbf{x}}^l(\hat{\mathbf{x}}^{l+1})^T - N_g\hat{\mathbf{x}}^{l+1}(\hat{\mathbf{x}}^l)^T\right).\end{aligned} \tag{35}$$

By using (35) instead of (34), the slowest step in the computation of the covariance matrices, which is the computation of $\dot{\mathbf{C}}_{\text{ser}}$, can be reduced in complexity from $O(L(N_g)^2)$ to only $O(N)$ operations ($N = LN_g$), which is of the same order as the computation of $\mathbf{C}_{\text{ser}}$. Thus, for the same number of samples N, we obtain a larger speed-up for larger group sizes.

Discretization introduces some type of quantization error. While a large number of discrete labels L results in a smaller quantization error, having too many of them is undesired because fewer edges would be considered, which would increase the number of samples needed to reduce the overall error. For example, in the extreme case of $N_g = 1$ and $L = N$, this method does not bring any benefit because it is almost equivalent to the sample reordering approach (differing only due to the smaller weights of the first and last samples).

5.4 Mixed Training Graph

The serial training graph does not have intra-group connections, and therefore the output differences of samples with the same label are not explicitly being minimized. One argument against intra-group connections is that if two vertices are adjacent to the same set of vertices, their corresponding samples are already likely to be mapped to similar outputs. However, in some cases, particularly for small numbers of training samples, additional intra-group connections might indeed improve robustness. We thus conceived the *mixed* training graph (Figure 9), which is a combination of the serial and clustered training graph and fulfils the consistency restriction (20). In the mixed training graph, all nodes and edges have a weight of 1, except for the intra-group edges in the first and last groups, which have a weight of 2. As expected, the computation of the covariance matrices can also be done efficiently for this training graph (details omitted).



$$v_n^l = 1, \text{ for } 1 \le n \le N_g \text{ and } 1 \le l \le L$$

$$\left.\begin{aligned} &\gamma_{n,n'}^{1,1} = 2 \\ &\gamma_{n,n'}^{l,l} = 1, \text{ for } 2 \le l \le L-1 \\ &\gamma_{n,n'}^{L,L} = 2 \end{aligned}\right\} \text{ (intra-group connections)}$$

$$\gamma_{n,n'}^{l,l+1} = 1, \text{ for } 1 \le l \le L-1 \text{ (inter-group connections)}$$

$$(1 \le n, n' \le N_g)$$

Figure 9: Illustration of the mixed training graph. Samples having the same label are fully connected (intra-group connections, represented with vertical edges) and all samples of adjacent groups are connected (inter-group connections). All vertex and edge weights are equal to 1 except for the intra-group edge weights of the first and last groups, which are equal to 2 as represented by thick lines.

5.5 Supervised Step for Regression Problems

There are at least three approaches to implement the supervised step on top of SFA to learn a mapping from slow features to the labels. The first one is to use a method such as linear or nonlinear regression. The second one is to discretize the original labels to a small discrete set $\{\tilde{\ell}_1, \ldots, \tilde{\ell}_{\tilde{L}}\}$ (which might be different from the discrete set used by the training graphs). The discrete labels are then treated as classes, and a classifier is trained to predict them from the slow features. One can then output the predicted class as the estimated label. Of course, an error due to the discretization of the labels is unavoidable. The third approach improves on the second one by using a classifier that also estimates class membership probabilities. Let $\mathbf{P}(C_{\tilde{\ell}_l}|\mathbf{y})$ be the estimated class probability

that the input sample $\mathbf{x}$ with slow features $\mathbf{y} = \mathbf{g}(\mathbf{x})$ belongs to the group with (discretized) label $\tilde{\ell}_l$. Class probabilities can be used to provide a more robust estimation of a soft (continuous) label ℓ, better suited to the particular loss function. For instance, one can use

$$\ell \stackrel{\text{def}}{=} \sum_{l=1}^{\tilde{L}} \tilde{\ell}_l \cdot \mathbf{P}(C_{\tilde{\ell}_l}|\mathbf{y}) \tag{36}$$

if the loss function is the RMSE, where the slow features $\mathbf{y}$ might be extracted using any of the four SFA-based methods for regression above. Other loss functions, such as the Mean Average Error (MAE), can be addressed in a similar way.

We have tested these three approaches in combination with supervised algorithms such as linear regression, and classifiers such as nearest neighbour, nearest centroid, Gaussian classifier, and SVMs. We recommend using the soft labels computed from the class probabilities estimated by a Gaussian classifier because in most of our experiments this method has provided best performance and robustness. Of course, other classifiers providing class probabilities could also be used.

6. Experimental Evaluation of the Methods Proposed

In this section, we evaluate the performance of the supervised learning methods based on SFA presented above. We consider two concrete image analysis problems using real photograph databases, the first one for classification and the second one for regression.

6.1 Classification

For classification, we have proposed the clustered training graph. As mentioned in Section 4.2, when this graph is used, the outputs of GSFA are equivalent to those of FDA. Since FDA has been used and evaluated exhaustively, here we only verify that our implementation of GSFA generates the expected results when trained with such a graph.

The German Traffic Sign Recognition Benchmark (Stallkamp et al., 2011) was chosen for the experimental test. This was a competition with the goal of classifying photographs of 43 different traffic signs taken on German roads under uncontrolled conditions with variations in lighting, sign size, and distance. No detection step was necessary because the true position of the signs was included as annotations, making this a pure classification task and ideal for our test. We participated in the online version of the competition, where 26,640 labeled images were provided for training and 12,569 images without label for evaluation (classification rate was computed by the organisers, who had ground-truth data).

Two-layer nonlinear cascaded (non-hierarchical) SFA was employed. To achieve good performance, the choice of the nonlinear expansion function is crucial. If it is too simple (e.g., low-dimensional), it does not solve the problem; if it is too complex (e.g., high-dimensional), it might overfit to the training data and not generalize well to test data. In all the experiments done here, a compact expansion that only doubles the data dimension was employed, $\mathbf{x}^T \mapsto \mathbf{x}^T, (|\mathbf{x}|^{0.8})^T$, where the absolute value and exponent 0.8 are computed component-wise. We refer to this expansion as $0.8Exp$. Previously, Escalante-B. and Wiskott (2011) have reported that it offers good generalization and competitive performance in SFA networks, presumably due to its robustness to outliers and certain properties regarding the approximation of higher frequency harmonics.

Our method, complemented by a Gaussian classifier on 42 slow features, achieved a recognition rate of 96.4% on test data.[3] This, as expected, was similar to the reported performance of various methods based on FDA participating in the same competition. For comparison, human performance was 98.81%, and a convolutional neural network gave top performance with a 98.98% recognition rate.

6.2 Regression

The remaining training graphs have all been designed for regression problems and were evaluated with the problem of estimating the horizontal position of a face in frontal face photographs, an important regression problem because it can be used as a component of a face detection system, as we proposed previously (see Mohamed and Mahdi, 2010). In our system, face detection is decomposed into the problems of the estimation of the horizontal position of a face, its vertical position, and its size. Afterwards, face detection is refined by locating each eye more accurately with the same approach applied now to the eyes instead of to the face centers. Below, we explain this regression problem, the algorithms evaluated, and the results in more detail.

6.2.1 PROBLEM AND DATA SET DESCRIPTION

To increase image variability and improve generalization, face images from several databases were used, namely 1,521 images from BioID (Jesorsky et al., 2001), 9,030 from CAS-PEAL (Gao et al., 2008), 5,479 from Caltech (Fink et al.), 9,113 from FaceTracer (Kumar et al., 2008), and 39,328 from FRGC (Phillips et al., 2005) making a total of 64,471 images, which were automatically pre-processed through a pose-normalization and a pose-reintroduction step. In the first step, each image was converted to greyscale and pose-normalized using annotated facial points so that the face is centered,[4] has a fixed eye-mouth-triangle area, and the resulting pose-normalized image has a resolution of 256×192 pixels. In the second step, horizontal and vertical displacements were re-introduced, as well as scalings, so that the center of the face deviates horizontally at most ±45 pixels from the center of the image. The vertical position and the size of the face were randomized, so that vertically the face center deviates at most ±20 pixels, and the smallest faces are half the size of the largest faces (a ratio of at most 1 to 4 in area). Interpolation (e.g., needed for scaling and sub-pixel displacements) was done using bicubic interpolation. At last, the images were cropped to 128×128 pixels.

Given a pre-processed input image, as described above, with a face at position (x,y) w.r.t. the image center and size z, the regression problem is then to estimate the x-coordinate of the center of the face. The range of the variables x,y and z is bounded to a box, so that one does not have to consider extremely small faces, for example. To assure a robust estimation for new images, invariance to a large number of factors is needed, including the vertical position of the face, its size, the expression and identity of the subject, his or her accessories, clothing, hair style, the lighting conditions, and the background.

3. Interestingly, GSFA did not provide best performance directly on the pixel data, but on precomputed HOG features. Ideally, pre-processing is not needed if SFA has an unrestricted feature space. In practice, knowing a good low-dimensional set of features for the particular data is beneficial. Applying SFA to such features, as commonly done with other machine learning algorithms, can reduce overfitting.

4. The center of a face was defined here as $\frac{1}{4}\mathbf{LE}+\frac{1}{4}\mathbf{RE}+\frac{1}{2}\mathbf{M}$, where **LE**, **RE** and **M** are the coordinates of the centers of the left eye, right eye and mouth, respectively. Thus, the face center is the midpoint between the mouth and the midpoint of the eyes.

Figure 10: Example of a pose-normalized image (left), and various images after pose was reintroduced illustrating the final range of vertical and horizontal displacements, as well as the face sizes (right).

The pose-normalized images were randomly split in three data sets of $30,000$, $20,000$ and $9,000$ images. The first data set was used to train the dimensionality reduction method, the second one to train the supervised post-processing step, and the last one for testing. To further exploit the images available, the pose-normalized images of each data set were duplicated, resulting in two pose-reintroduced images per input image, that is, a single input image exclusively belongs to one of the three data sets, appearing twice in it with two different poses. Hence, the final size of the data sets is $60,000$, $40,000$ and $18,000$ pre-processed images, respectively.

6.2.2 DIMENSIONALITY-REDUCTION METHODS EVALUATED

The resolution of the images and their number make it less practical to directly apply SFA and the majority of supervised methods, such as an SVM, and unsupervised methods, such as PCA/ICA/LLE, to the raw images. We circumvent this by using three efficient dimensionality reduction methods, and by applying supervised processing on the lower-dimensional features extracted. The first two methods are efficient hierarchical implementations of SFA and GSFA (referred to as HSFA without distinction). The nodes in the HSFA networks first expand the data using the $0.8Exp$ expansion function (see Section 6.1) and then apply SFA/GSFA to it, except for the nodes in the first layer in which additionally PCA is applied before the expansion preserving 13 out of 16 principal components. For comparison, we use a third method, a hierarchical implementation of PCA (HPCA), in which all nodes do pure PCA. The structure of the hierarchies for the HSFA and PCA networks is described in Table 1. In contrast to other works (e.g., Franzius et al., 2007), weight-sharing was not used at all, improving feature specificity at the lowest layers. The input to the nodes (fan-in) comes mostly from two nodes in the previous layer. This small fan-in reduces the computational cost because the input dimensionality is minimized. This also results in networks with a large number of layers potentiating the accumulation of non-linearity across the network. Non-overlapping receptive fields were used because in previous experiments with similar data they showed good performance at a smaller computational cost.

The following dimensionality-reduction methods were evaluated (one based on SFA, four based on GSFA, and one based on PCA).

- SFA using sample reordering (reordering).
- GSFA with a mirrored sliding window graph with $d = 32$ (MSW32).

Layer	size	node fan-in	output dim. per HSFA node	output dim. per HPCA node
0 (input image)	128×128 pixels	—	—	—
1	32×32 nodes	4×4	13	13
2	16×32 nodes	2×1	20	20
3	16×16 nodes	1×2	35	35
4	8×16 nodes	2×1	60	60
5	8×8 nodes	1×2	60	100
6	4×8 nodes	2×1	60	120
7	4×4 nodes	1×2	60	120
8	2×4 nodes	2×1	60	120
9	2×2 nodes	1×2	60	120
10	1×2 nodes	2×1	60	120
11 (top node)	1×1 nodes	1×2	60	120

Table 1: Structure of the SFA and PCA deep hierarchical networks. The networks only differ in the type of processing done by each node and in the number of features preserved. For HSFA an upper bound of 60 features was set, whereas for HPCA at most 120 features were preserved. A node with a fan-in of $a \times b$ is driven by a rectangular array of nodes (or pixels for the first layer) with such a shape, located in the preceding layer.

- GSFA with a mirrored sliding window graph with $d = 64$ (MSW64).
- GSFA with a serial training graph with $L = 50$ groups of $N_g = 600$ images (serial).
- GSFA with a mixed graph and the same number of groups and images (mixed).
- A hierarchical implementation of PCA (HPCA).

It is impossible to compare GSFA against all the dimensionality reduction and supervised learning algorithms available, and therefore we made a small selection thereof. We chose HPCA for efficiency reasons and because it is likely to be a good dimensionality reduction algorithm for the problem at hand since principal components code well the coarse structure of the image including the silhouette of the subjects, allowing for a good estimation of the position of the face. Thus, we believe that HPCA (combined with various supervised algorithms) is a fair point of comparison, and a good representative among generic machine learning algorithms for this problem. For the data employed, 120 HPCA features at the top node explain 88% of the data variance, suggesting that HPCA is indeed a good approximation to PCA in this case.

The evolution across the hierarchical network of the two slowest features extracted by HSFA is illustrated in Figure 11.

6.2.3 SUPERVISED POST-PROCESSING ALGORITHMS CONSIDERED

On top of the dimensionality reduction methods, we employed the following supervised post-processing algorithms.

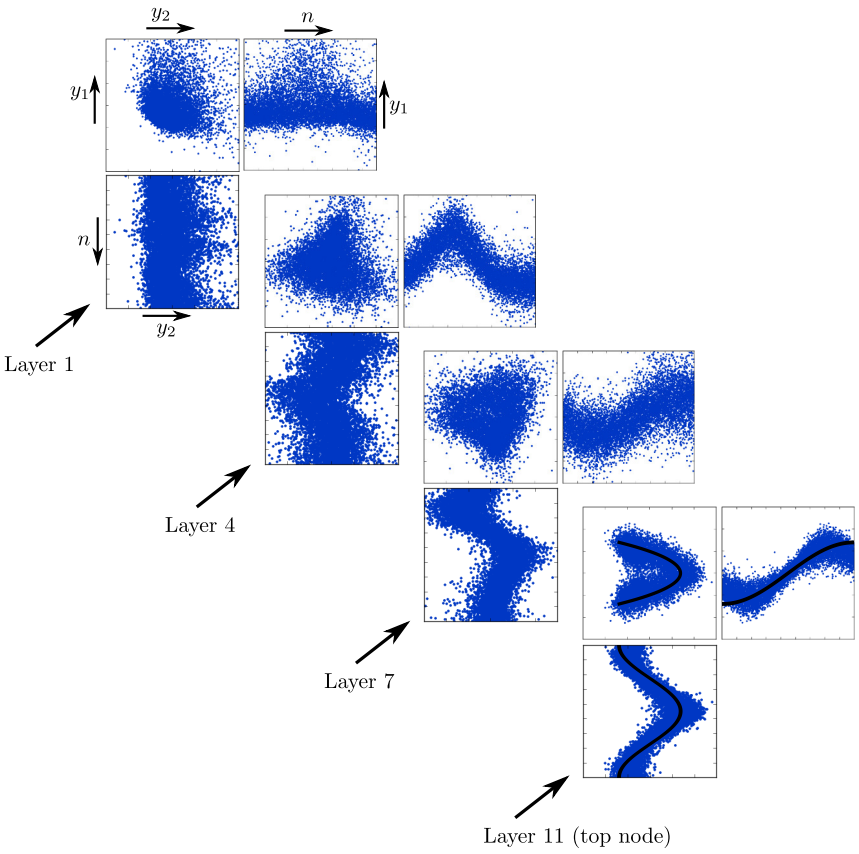


Figure 11: Evolution of the slow features extracted from test data after layers 1, 4, 7 and 11 of a GSFA network trained with the serial training graph. A central node was selected from each layer, and three plots are provided, that is, y_2 vs y_1, n vs y_1, and y_2 vs n. Hierarchical processing results in progressively slower features as one moves from the first to the top layer. The solid line in the plots of the top node represents the optimal free responses, which are the slowest possible features one can obtain using an unrestricted mapping, as predicted theoretically (Wiskott, 2003). Notice how the features evolve from being mostly unstructured in the first layer to being similar to the free responses at the top node, indicating success at finding the hidden parameter changing most slowly for these data (i.e., the horizontal position of the faces).

- A nearest centroid classifier (NCC).
- Labels estimated using (36) and the class membership probabilities given by a Gaussian classifier (Soft GC).
- A multi-class (one-versus-one) SVM (Chang and Lin, 2011) with a Gaussian radial basis kernel, and grid search for model selection.
- Linear regression (LR).

To train the classifiers, the images of the second data set were grouped in 50 equally large classes according to their horizontal displacement x, $-45 \leq x \leq 45$.

6.2.4 RESULTS

We evaluated all the combinations of a dimensionality reduction method (reordering, MSW32, MSW64, serial, mixed and HPCA) and a supervised post-processing algorithm (NCC, Soft GC, SVM, LR). Their performance was measured on test data and reported in terms of the RMSE. The labels estimated depend on three parameters: the number of features passed to the supervised post-processing algorithm, and the parameters C and γ in the case of the SVM. These parameters were determined for each combination of algorithms using a single trial, but the RMSEs reported here were averaged over 5 trials.

The results are presented in Table 2, and analyzed focusing on four aspects: the dimensionality-reduction method, the number of features used, the supervised methods, and the training graphs. For any choice of the post-processing algorithm and training graph, GSFA resulted in an RMSE 5% to 13% smaller than when using the basic reordering of samples employing standard SFA. In turn, reordering resulted in an RMSE at least 10% better for this data set than when using HPCA.

Dim. reduction method	**NCC** (RMSE)	# of feat.	**Soft GC** (RMSE)	# of feat.	**SVM** (RMSE)	# of feat.	**LR** (RMSE)	# of feat.
Reordering/SFA	6.16	6	5.63	4	6.00	14	10.23	60
MSW32 (GSFA)	5.78	5	5.25	4	5.52	18	9.74	60
MSW64 (GSFA)	5.69	5	5.15	4	5.38	18	9.69	60
Serial (GSFA)	**5.58**	4	**5.03**	5	**5.23**	15	9.68	60
Mixed (GSFA)	5.63	4	5.12	4	5.40	19	**9.54**	60
HPCA	29.68	118	6.17	54	8.09	50	19.24	120

Table 2: Performance (RMSE) of the dimensionality reduction algorithms measured in pixels in combination with various supervised algorithms for the post-processing step. The RMSE at chance level is 25.98 pixels. Each entry reports the best performance achievable using a different number of features and parameters in the post-processing step. Largest standard deviation of 0.21 pixels. Clearly, linear regression benefited from all the SFA and PCA features available.

Taking a look at the number of features used by each supervised post-processing algorithm, one can observe that considerably fewer HSFA-features are used than HPCA-features (e.g., 5 vs. 54 for Soft GC). This can be explained because PCA is sensitive to many factors that are irrelevant to solve the regression problem, such as the vertical position of the face, its scale, the background, lighting, etc. Thus, the information that encodes the horizontal position of a face is mixed with other information and distributed over many principal components, whereas it is more concentrated in the slowest components of SFA.

If one focuses on the post-processing methods, one can observe that linear regression performed poorly confirming that a linear supervised step is too weak, particularly when the dimensionality reduction is also linear (e.g., HPCA). The nearest centroid classifier did modestly for HSFA, but

even worse than the chance level for HPCA. The SVMs were consistently better, but the error can be further reduced by 4% to 23% by using Soft GC, the soft labels derived from the Gaussian classifier.

Regarding the training graphs, we expected that the sliding window graphs, MSW32 and MSW64, would be more accurate than the serial and mixed graphs, even when using a square window, because the labels are not discretized. Surprisingly, the mixed and serial graphs were the most accurate ones. This might be explained in part by the larger number of connections in these graphs. Still, MSW32 and MSW64 were better than the reordering approach, the wider window being superior. The RMSE of the serial graph was smaller than the one of the mixed graph by less than 2% (for Soft GC), making it uncertain for statistical reasons which one of these graphs is better for this problem. A larger number of trials, or even better, a more detailed mathematical analysis of the graphs might be necessary to determine which one is better.

7. Discussion

In this paper, we propose the graph-based SFA (GSFA) optimization problem, an extension of the standard SFA optimization problem that takes into account the information contained in a structure called training graph, in which the vertices are the training samples and the edges represent connections between samples. Edge weights allow the specification of desired output similarities and can be derived from label or class information. The GSFA optimization problem generalizes the notion of slowness defined originally for a plain sequence of samples to such a graph.

We also propose the GSFA algorithm, an implicitly supervised extension of the (unsupervised) SFA algorithm, and prove that GSFA solves the new optimization problem in the function space considered. The main goal of GSFA is to solve supervised learning problems by reducing the dimensionality of the data to a few very label-predictive features.

We call GSFA implicitly supervised because the labels themselves are never provided to it, but only the training graphs, which encode the labels through their structure. While the construction of the graph is a supervised operation, GSFA works in an unsupervised fashion on structured data. Hence, GSFA does not search for a fit to the labels explicitly but instead fully concentrates on the generation of slow features according to the topology defined by the graph.

Several training graphs for classification or regression are introduced in this paper. We have designed them aiming at a balance between speed and accuracy. These graphs offer a significant advantage in terms of speed, for example, over other similarity matrices typically used with LPP. Conceptually, such a speed-up can be traced back to two factors that originate from the highly regular structure of the graphs (Sections 4 and 5). First, determining the edges and edge weights is a trivial operation because they are derived from the labels in a simple manner. In contrast, this operation can be quite expensive if the connections are computed using nearest neighbour algorithms. Second, as we have shown, linear algebra can be used to optimize the computation of $\dot{\mathbf{C}}$, which is needed during the training phase. The resulting complexity for training is linear in the number of samples, even though the number of connections considered is quadratic. The experimental results demonstrate that the larger number of connections considered by GSFA indeed provides a more robust learning than standard SFA, making it superior to SFA in supervised learning settings.

When solving a concrete supervised learning problem, the features extracted by unsupervised dimensionality reduction algorithms are often suboptimal. For instance, PCA does not yield good features for age estimation from adult face photographs because features revealing age (e.g., skin

textures) have higher spatial frequencies and do not belong to the main principal components. Supervised dimensionality reduction algorithms, including GSFA, are specially promising when one does not know a good set of features for the particular data and problem at hand, and one wants to improve performance by generating features adapted to the specific data and labels.

One central idea of this paper, shown in Figure 1 (left), is the following. If one has a large number of high-dimensional labeled data, supervised learning algorithms can often not be applied due to high computational requirements. In such cases we suggest to transform the labeled data to structured data, where the label information is implicitly encoded in the connections between data points. Then, unsupervised learning algorithms, such as SFA, or its implicitly supervised extension GSFA, can be used. This permits hierarchical processing for dimensionality reduction, an operation that is frequently more difficult with supervised learning algorithms. The resulting low-dimensional data has an intrinsic structure that reflects the graph topology. These data can then be transformed back to labeled data by adding the labels, and standard supervised learning algorithms can be applied to solve the original supervised learning problem.

7.1 Related Optimization Problems and Algorithms

Recently, Böhmer et al. (2012) introduced regularized sparse kernel SFA. The algorithm was applied to solve a classification problem by reducing the data dimensionality. In the discussion section, various extensions similar to the GSFA optimization problem were briefly presented without empirical evaluation. For classification, the authors propose an objective function equivalent to (6), with edge weights $\gamma_{n,n'} = \delta_{c_n c_{n'}}$, where c_n and $c_{n'}$ are the classes of the respective samples, and $\delta_{c_n c_{n'}}$ is the Kronecker delta. If all classes $C_1, \ldots, C_S$ are equally represented by N_s samples, such edge weights are equivalent to those specified by the clustered graph. However, if N_s is not the same for all classes, the binary edge weights (either 1 or 0, as given by the Kronecker delta) are less appropriate in our view because larger classes are overrepresented by the quadratic number of edges $N_s(N_s+1)/2$ for class s. The authors also consider transitions with variable weights. For this purpose, they use an importance measure $p(\mathbf{x}_{t+1}, \mathbf{x}_t) \geq 0$ with high values for transitions within the desired subspace and propose the objective function

$$\min s'(y_i) \stackrel{\text{def}}{=} \frac{1}{n-1} \sum_{t=1}^{n-1} \frac{(y_i(t+1) - y_i(t))^2}{p(\mathbf{x}_{t+1}, \mathbf{x}_t)},$$

with $y_i(t) \stackrel{\text{def}}{=} \phi_i(\mathbf{x}(t))$. This accounts for arbitrary edge weights $\gamma_{n,n+1}$ in a linear graph, which could be easily generalized to arbitrary graphs. It is not clear to us why the importance measure p has been introduced as a quotient instead of as a factor. The authors also propose an importance measure $q(\mathbf{x})$ for the samples, which plays exactly the same role as the node weights $\{v_n\}_n$. The unit variance and decorrelation constraints are adapted to account for $q(\mathbf{x})$ and become fully equivalent to constraints (8–9) of GSFA. The remaining zero-mean constraint was not explicitly adapted.

Zhang et al. (2009) propose a framework for the systematic analysis of several dimensionality reduction algorithms, such as LLE, LE, LPP, PCA and LDA, to name just a few. Such a framework is useful for comparing linear SFA and GSFA to other algorithms from a mathematical point of view, regardless of their typical usage and application areas.

The authors show that LPP is a linearization of LE. In turn, linear SFA can be seen as a special case of LPP, where the weight matrix has a special form (see Sprekeler, 2011). Consider the

following LPP optimization problem (He and Niyogi, 2003):

$$\arg\min_{\substack{\mathbf{y} \\ \mathbf{y}^T D \mathbf{y}=1}} \frac{1}{2}\sum_{i,j}(y_i - y_j)^2 W_{ij} = \mathbf{y}^T \mathbf{L}\mathbf{y},$$

where $\mathbf{W}$ is a symmetric weight matrix, $\mathbf{D}$ is a diagonal matrix with diagonal $D_{ii} \stackrel{\text{def}}{=} \sum_j W_{ij}$, $\mathbf{L} \stackrel{\text{def}}{=} \mathbf{D} - \mathbf{W}$ is the Laplacian matrix, and the output features $\mathbf{y} \stackrel{\text{def}}{=} \mathbf{a}^T\mathbf{x}$ are linear in the input. The equivalence to linear SFA is achieved if one sets $\mathbf{W}$ as $W_{ij} = \frac{1}{2}(\delta_{i,1}\delta_{j,1} + \delta_{i,N}\delta_{j,N} + \delta_{j,i+1} + \delta_{i,j+1})$. The objective function then becomes the same as in SFA, and $\mathbf{D}$ becomes the identity matrix, yielding the same restrictions as in SFA. The zero-mean constraint is implicit and achieved by discarding a constant solution with eigenvalue zero. Notice that leaving out the terms $\delta_{i,1}\delta_{j,1} + \delta_{i,N}\delta_{j,N}$ results in $\mathbf{D} = \text{diag}(1/2, 1, 1, \ldots, 1, 1/2)$ being slightly different from the identity and the equivalence would only be approximate.

Sprekeler (2011) studied the relation between SFA and LE and proposed the combination of the neighbourhood relation used by LE and the functional nature of SFA. The resulting algorithm, called generalized SFA, computes a functional approximation to LE with a feature space spanned by a set of basis functions. Linear generalized SFA is equivalent to linear LPP (Rehn, 2013), and in general it can be regarded as LPP on the data expanded by such basis functions.

Also the strong connection between LPP and linear GSFA is evident from the optimization problem above. In this case, the elements D_{ii} play the role of the node weights v_i, and the elements W_{ij} play the role of the edge weights $\gamma_{i,j}$. One difference between LPP and GSFA is that the additional normalization factors Q and R of GSFA provide invariance to the scale of the edge and node weights specifying a particular feature and objective function normalization. A second difference is that for GSFA the node weights, fundamental for feature normalization, can be chosen independently from the edge weights (unless one explicitly enforces (20)), whereas for LPP the diagonal elements $D_{ii} \stackrel{\text{def}}{=} \sum_j W_{ij}$ are derived from the edge weights.

We show now how one can easily compute LPP features using GSFA. Given a similarity matrix W_{ij} and diagonal matrix $\mathbf{D}$ with diagonal elements $D_{ii} \stackrel{\text{def}}{=} \sum_j W_{ij}$, one solves a GSFA problem defined over a training graph with the same samples, edge weights $\gamma_{i,j} = W_{ij}$, and node weights $v_i = D_{ii}$. The optimization problem solved by GSFA is then equivalent to the LPP problem, except for the scale of the objective function and the features. If the features extracted from a particular sample are denoted as $\mathbf{y}_{\text{GSFA}}$, one can match the feature scales of LPP simply by applying a scaling $\mathbf{y} = Q^{-1/2}\mathbf{y}_{\text{GSFA}}$, where $Q \stackrel{(11)}{=} \sum_i v_i$.

It is also possible to use LPP to compute GSFA features. Given a training graph with edge weights $\gamma_{i,j}$ and node weights v_i, we can define the following similarity matrix:

$$W_{ij} = \begin{cases} 2\gamma_{i,j}/R, & \text{for } i \neq j, \text{ with } R \text{ as defined in (10)}, \\ v_i/Q - \sum_{j' \neq i} W_{ij'}, & \text{for } i = j, \text{ with } Q \text{ as defined in (11)}. \end{cases}$$

The similarity matrix $\mathbf{W}$ above ensures the same objective function as in GSFA, and also that $D_{ii} \stackrel{\text{def}}{=} \sum_j W_{ij} = v_i/Q$, resulting in the same constraints. In practice, using self-loops $W_{ii} \neq 0$ is unusual for LPP, but they are useful in this case.

7.2 Remarks on Classification with GSFA

Both classification and regression tasks can be solved with GSFA. We show that a few slow features allow for an accurate solution to the supervised learning problem, requiring only a small post-processing step that maps the features to labels or classes.

For classification problems, we propose a clustered training graph, which yields features having the discrimination capability of FDA. The results of the implementation of this graph confirm the expectations from theory. Training with the clustered graph is equivalent to considering all transitions between samples of the same identity and no transition between different identities. The computation, however, is more efficient than the direct method of Berkes (2005a), where a large number of transitions have to be considered explicitly.

The Markov chain generated through the probabilistic interpretation of this graph is equal to the Markov chain defined by Klampfl and Maass (2010). These Markov chains are parameterized by vanishing parameters ε and a, respectively. The parameter a influences the probability $P_{ij} = aN_j/N$ of transitioning from a class c_i to a different class c_j, where N_j is the number of samples of class c_j and N is the total number of samples. Thus, in the limit $a \to 0$ all transitions lie within the same class. However, the Markov chain and ε are introduced here for analytical purposes only. In practice, GSFA directly uses the graph structure, which is deterministic and free of ε. This avoids the less efficient training of SFA with a very long sequence of samples generated with the Markov chain, as done by Klampfl and Maass (2010). (Even though the set of samples is finite, as the parameter a approaches 0, an infinite sequence is required to properly capture the data statistics of all identities).

Klampfl and Maass (2010) have proven that if $a \to 0$ the features learned by SFA from the data generated are equivalent to the features learned by FDA. From the equality of the two Markov chains above, the features extracted by GSFA turn out to be also equivalent to those of FDA. Thus, the features extracted with GSFA from the clustered graph are not better or worse than those extracted with FDA. However, this equivalence is an interesting result because it allows a different interpretation of FDA from the point of view of the generation of slow signals. Moreover, advances in generic methods for SFA, such as hierarchical network architectures or robust nonlinearities, result in improved classification rates over standard FDA.

It is possible to design other training graphs for classification without the equivalence to FDA, for example, by using non-constant sample weights, or by incorporating input similarity information or other criteria in the edge-weight matrix. This idea has been explored by Rehn (2013) using generalized SFA, where various adjacency graphs (i.e., edge weights) were proposed for classification inspired by the theory of manifold learning. Instead of using full-class connectivity, only samples within the same class among the first nearest neighbours are connected. Less susceptibility to outliers and better performance have been reported.

7.3 Remarks on Regression with GSFA

To solve regression problems, we propose three training graphs for GSFA that resulted in a reduction of the RMSE of up to 11% over the basic reordering approach using standard SFA, an improvement caused by the higher number of similarity relations considered even though the same number of training samples is used.

First extensions of SFA for regression (i.e., GSFA) were employed by Escalante-B. and Wiskott (2010) to estimate age and gender from frontal static face images of artificial subjects, created with

special software for 3D face modelling and rendering. Gender estimation was treated as a regression problem, because the software represents gender as a continuous variable (e.g., -1.0=typical masculine face, 0.0=neutral, 1.0=typical feminine face). Early, undocumented versions of the mixed and serial training graphs were employed. Only three features extracted were passed to an explicit regression step based on a Gaussian classifier (Section 5.5). In both cases, good performance was achieved, with an RMSE of 3.8 years for age and 0.33 units for gender on test data, compared to a chance level of 13.8 years and 1.73 units, respectively.

With a system similar to the one presented here, we participated in a face detection competition successfully (Mohamed and Mahdi, 2010). We estimated the x-position, y-position and scale of a face within an image. Using three separate SFA networks, one for each parameter, faces can be pose-normalized. A fourth network can be trained to estimate the quality of the normalization, again as a continuous parameter, and to indicate whether a face is present at all. These four networks together were used to detect faces. Performance of the resulting face detection system on various image databases was competitive and yielded a detection rate on greyscale photographs within the range from 71.5% to 99.5% depending on the difficulty of the test images. An increase in the image variability in the training data can be expected to result in further improvements.

The serial and mixed training graphs have provided the best accuracy for the experiments in this article, with the serial one being slightly more accurate but not to a significant level. These graphs have the advantage over the sliding window graph that they are also suitable for cases in which the labels are discrete from the beginning, which occurs frequently due to the finite resolution in measurements or due to discrete phenomena (e.g., when learning the number of red blood cells in a sample, or a distance in pixels).

Since the edge weights supported by GSFA are arbitrary, it is tempting to use a complete weight matrix continuous in the labels (e.g., $\gamma_{n,n'} = \frac{1}{|\ell_{n'} - \ell_n| + k}$, for $k > 0$, or $\gamma_{n,n'} = \exp(-\frac{(\ell_{n'} - \ell_n)^2}{\sigma^2})$). However, this might affect the training time markedly. Moreover, one should be aware that imbalances in the connectivity of the samples might result in pathological solutions or less useful features than expected.

In this work, we have focused on supervised dimensionality reduction towards the estimation of a single label. However, one can estimate two or more labels simultaneously using appropriate training graphs. In general, using such graphs might reduce the performance of the method compared to the separate estimation of the labels. However, if the labels are intrinsically related, performance might actually improve. For instance, using another algorithm, the estimation of age from face images has been reported to improve when also gender and race labels are estimated (Guo and Mu, 2011). This is justified because gender and race are two factors that strongly influence the aging process.

The features extracted by GSFA strongly depend on the labels, even though label information is only provided implicitly by the graph connectivity. Ideally, the slowest feature extracted is a monotonic function of the hidden label, and the remaining features are harmonics of increasing frequency of the first one. In practice, noisy and distorted versions of these features are found, but still providing an approximate, redundant, and concentrated coding of the labels. Their simple structure permits the use of simple supervised algorithms for the post-processing step saving time and computer resources. For the estimation of the x-position of faces, all the nonlinear post-processing algorithms, including the nearest centroid classifier, provided good accuracy. Although a Gaussian classifier is a less powerful classifier than an SVM, the estimation based on the class membership

probabilities of the Gaussian classifier (Soft GC) is more accurate because it reduces the effect of miss-classifications.

7.4 Other Considerations

Locality preserving projections and GSFA come from very different backgrounds. On the one hand, LPP is motivated from unsupervised learning and was conceived as a linear algorithm. The similarity matrices used are typically derived from the training samples, for example, using a heat kernel function. Later, weight matrices accounting for label information have been proposed, particularly for classification. On the other hand GSFA is motivated from supervised learning, and was conceived as an extension of SFA designed for supervised non-linear dimensionality reduction specifically targeting regression and classification problems. Although the motivation behind LPP and GSFA, as well as their typical applications, are different, these algorithms are strongly connected. Therefore, it might be worth not only to unify their formalism, but also the conceptual roots that have inspired them.

Although supervised learning is less biologically plausible, GSFA being implicitly supervised is still closely connected to feasible unsupervised biological models through the probabilistic interpretation of the graph. If we ensure that the graph fulfils the normalization restrictions, the Markov chain described in Section 3.5 can be constructed, and learning with GSFA and such graph becomes equivalent to learning with standard (unsupervised) SFA as long as the training sequence originates from the Markov chain. From this perspective, GSFA uses the graph information to simplify a learning procedure that could also be done unsupervised.

We do not claim that GSFA is better or worse than other supervised learning algorithms, it is actually equivalent to other algorithms under specific conditions. We only show that it is better for supervised learning than SFA, and believe that it is a very interesting and promising algorithm. Of course, specialized algorithms might outperform GSFA for particular tasks. For instance, algorithms for face detection can outperform the system presented here, but a fundamental advantage of GSFA is that it is general purpose. Moreover, various improvements to GSFA (not discussed here) are under development, which will increase its performance and narrow the gap to special-purpose algorithms.

One limitation of hierarchical processing with GSFA or SFA (i.e., HSFA) is that the features should be spatially localized in the input data. For instance, if one randomly shuffles the pixels in the input image, performance would decrease considerably. This limits the applicability of HSFA for scenarios with heterogeneous independent sources of data, but makes it well suited, for example, for images.

Although GSFA makes a more efficient use of the samples available than SFA, it can still overfit in part because these algorithms lack an explicit regularization parameter. Hence, for a small number of samples data randomization techniques are useful. Interestingly, certain expansions and hierarchical processing can be seen as implicit regularization measures. In fact, less overfitting compared to standard SFA is one of the central advantages of using HSFA. A second central advantage of HSFA is that HSFA networks can be trained in a feed-forward manner layer by layer, resulting in a very efficient training. Due to accumulation of nonlinearities across the network, the features at the top node can be highly nonlinear w.r.t. the input, potentially spanning a rich feature space.

Most of this work was originally motivated by the need to improve generalization of our learning system. Of course, if the amount of training data and computation time were unrestricted, overfit-

ting would be negligible, and all SFA training methods would approximately converge to the same features and provide similar performance. For finite data and resources, the results demonstrate that GSFA does provide better performance than SFA (reordering method) using the same amount of training data. Another interpretation is that GSFA demands less training data to achieve the same performance, thus, indeed contributing to our pursuit of generalization.

Acknowledgments

We would like to thank Mathias Tuma for his helpful suggestions on an earlier version of this work and Fabian Schönfeld and Björn Weghenkel for their valuable corrections. The comments of the anonymous reviewers are also kindly appreciated. A. Escalante is jointly supported by the German Academic Exchange Service (DAAD) and the National Council of Science and Technology of Mexico (CONACYT).

References

T. Adali and S. Haykin. *Adaptive Signal Processing: Next Generation Solutions*. Adaptive and Learning Systems for Signal Processing, Communications and Control Series. John Wiley & Sons, 2010.

P. Berkes. Pattern recognition with slow feature analysis. Cognitive Sciences EPrint Archive (CogPrints), February 2005a. URL `http://cogprints.org/4104/`.

P. Berkes. Handwritten digit recognition with nonlinear fisher discriminant analysis. In *ICANN*, volume 3697 of *LNCS*, pages 285–287. Springer Berlin/Heidelberg, 2005b.

P. Berkes and L. Wiskott. Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 5(6):579–602, 2005.

W. Böhmer, S. Grünewälder, H. Nickisch, and K. Obermayer. Generating feature spaces for linear algorithms with regularized sparse kernel slow feature analysis. *Machine Learning*, 89(1):67–86, 2012.

A. Bray and D. Martinez. Kernel-based extraction of slow features: Complex cells learn disparity and translation invariance from natural images. In *NIPS*, volume 15, pages 253–260, Cambridge, MA, 2003. MIT Press.

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

A. N. Escalante-B. and L. Wiskott. Gender and age estimation from synthetic face images with hierarchical slow feature analysis. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 240–249, 2010.

A. N. Escalante-B. and L. Wiskott. Heuristic evaluation of expansions for Non-Linear Hierarchical Slow Feature Analysis. In *Proc. The 10th International Conference on Machine Learning and Applications*, pages 133–138, Los Alamitos, CA, USA, 2011.

A. N. Escalante-B. and L. Wiskott. Slow feature analysis: Perspectives for technical applications of a versatile learning algorithm. *Künstliche Intelligenz [Artificial Intelligence]*, 26(4):341–348, 2012.

M. Fink, R. Fergus, and A. Angelova. Caltech 10,000 web faces. URL `http://www.vision.caltech.edu/Image_Datasets/Caltech_10K_WebFaces/`.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7: 179–188, 1936.

P. Földiák. Learning invariance from transformation sequences. *Neural Computation*, 3(2):194–200, 1991.

M. Franzius, H. Sprekeler, and L. Wiskott. Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS Computational Biology*, 3(8):1605–1622, 2007.

M. Franzius, N. Wilbert, and L. Wiskott. Invariant object recognition with slow feature analysis. In *Proc. of the 18th ICANN*, volume 5163 of *LNCS*, pages 961–970. Springer, 2008.

M. Franzius, N. Wilbert, and L. Wiskott. Invariant object recognition and pose estimation with slow feature analysis. *Neural Computation*, 23(9):2289–2323, 2011.

W. Gao, B. Cao, S. Shan, X. Chen, D. Zhou, X. Zhang, and D. Zhao. The CAS-PEAL large-scale chinese face database and baseline evaluations. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 38(1):149–161, 2008.

G. Guo and G. Mu. Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 657–664, 2011.

X. He and P. Niyogi. Locality Preserving Projections. In *Neural Information Processing Systems*, volume 16, pages 153–160, 2003.

G. E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40(1-3):185–234, 1989.

O. Jesorsky, K. J. Kirchberg, and R. Frischholz. Robust face detection using the hausdorff distance. In *Proc. of Third International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 90–95. Springer-Verlag, 2001.

S. Klampfl and W. Maass. Replacing supervised classification learning by Slow Feature Analysis in spiking neural networks. In *Proc. of NIPS 2009: Advances in Neural Information Processing Systems*, volume 22, pages 988–996. MIT Press, 2010.

P. Koch, W. Konen, and K. Hein. Gesture recognition on few training data using slow feature analysis and parametric bootstrap. In *International Joint Conference on Neural Networks*, pages 1 –8, 2010.

T. Kuhnl, F. Kummert, and J. Fritsch. Monocular road segmentation using slow feature analysis. In *Intelligent Vehicles Symposium, IEEE*, pages 800–806, june 2011.

N. Kumar, P. N. Belhumeur, and S. K. Nayar. FaceTracer: A search engine for large collections of images with faces. In *European Conference on Computer Vision (ECCV)*, pages 340–353, 2008.

G. Mitchison. Removing time variation with the anti-hebbian differential synapse. *Neural Computation*, 3(3):312–320, 1991.

N. M. Mohamed and H. Mahdi. A simple evaluation of face detection algorithms using unpublished static images. In *10th International Conference on Intelligent Systems Design and Applications*, pages 1–5, 2010.

P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min, and W. Worek. Overview of the face recognition grand challenge. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 947–954, Washington, DC, USA, 2005. IEEE Computer Society.

E. M. Rehn. On the slowness principle and learning in hierarchical temporal memory. Master's thesis, Bernstein Center for Computational Neuroscience, 2013.

I. Rish, G. Grabarnik, G. Cecchi, F. Pereira, and G. J. Gordon. Closed-form supervised dimensionality reduction with generalized linear models. In *Proc. of the 25th ICML*, pages 832–839, New York, NY, USA, 2008. ACM.

H. Sprekeler. On the relation of slow feature analysis and laplacian eigenmaps. *Neural Computation*, 23(12):3287–3302, 2011.

H. Sprekeler and L. Wiskott. A theory of slow feature analysis for transformation-based input signals with an application to complex cells. *Neural Computation*, 23(2):303–335, 2011.

J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *International Joint Conference on Neural Networks*, pages 1453–1460, 2011.

M. Sugiyama. Local fisher discriminant analysis for supervised dimensionality reduction. In *Proc. of the 23rd ICML*, pages 905–912, 2006.

M. Sugiyama, T. Idé, S. Nakajima, and J. Sese. Semi-supervised local fisher discriminant analysis for dimensionality reduction. *Machine Learning*, 78(1-2):35–61, 2010.

W. Tang and S. Zhong. *Computational Methods of Feature Selection*, chapter Pairwise Constraints-Guided Dimensionality Reduction. Chapman and Hall/CRC, 2007.

R. Vollgraf and K. Obermayer. Sparse optimization for second order kernel methods. In *International Joint Conference on Neural Networks*, pages 145–152, 2006.

L. Wiskott. Learning invariance manifolds. In *Proc. of 5th Joint Symposium on Neural Computation, San Diego, CA, USA*, volume 8, pages 196–203. Univ. of California, 1998.

L. Wiskott. Slow feature analysis: A theoretical analysis of optimal free responses. *Neural Computation*, 15(9):2147–2177, 2003.

L. Wiskott and T. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.

D. Zhang, Z.-H. Zhou, and S. Chen. Semi-supervised dimensionality reduction. In *Proc. of the 7th SIAM International Conference on Data Mining*, 2007.

T. Zhang, D. Tao, X. Li, and J. Yang. Patch alignment for dimensionality reduction. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1299–1313, 2009.

Z. Zhang and D. Tao. Slow feature analysis for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):436 –450, 2012.

Journal of Machine Learning Research 14 (2013) 3721-3752 Submitted 4/11; Revised 5/13; Published 12/13

Joint Harmonic Functions and Their Supervised Connections

Mark Vere Culp MVCULP@MAIL.WVU.EDU
Kenneth Joseph Ryan KJRYAN@MAIL.WVU.EDU
Department of Statistics
West Virginia University
Morgantown, WV 26506, USA

Editor: Sridhar Mahadevan

Abstract

The cluster assumption had a significant impact on the reasoning behind semi-supervised classification methods in graph-based learning. The literature includes numerous applications where harmonic functions provided estimates that conformed to data satisfying this well-known assumption, but the relationship between this assumption and harmonic functions is not as well-understood theoretically. We investigate these matters from the perspective of supervised kernel classification and provide concrete answers to two fundamental questions. (i) Under what conditions do semi-supervised harmonic approaches satisfy this assumption? (ii) If such an assumption is satisfied then why precisely would an observation sacrifice its own supervised estimate in favor of the cluster? First, a harmonic function is guaranteed to assign labels to data in harmony with the cluster assumption if a specific condition on the boundary of the harmonic function is satisfied. Second, it is shown that any harmonic function estimate within the interior is a probability weighted average of supervised estimates, where the weight is focused on supervised kernel estimates near labeled cases. We demonstrate that the uniqueness criterion for harmonic estimators is sensitive when the graph is sparse or the size of the boundary is relatively small. This sets the stage for a third contribution, a new regularized joint harmonic function for semi-supervised learning based on a joint optimization criterion. Mathematical properties of this estimator, such as its uniqueness even when the graph is sparse or the size of the boundary is relatively small, are proven. A main selling point is its ability to operate in circumstances where the cluster assumption may not be fully satisfied on real data by compromising between the purely harmonic and purely supervised estimators. The competitive stature of the new regularized joint harmonic approach is established.

Keywords: harmonic function, joint training, cluster assumption, semi-supervised learning

1. Introduction

The problem under consideration is semi-supervised learning in the graph-based setting. Observations are vertices on a graph, and edges provide similarity associations between vertices. Classification is required if vertices are labeled and the goal is to design a function to predict the labels. Local classifiers like k-NN or more generally kernel regression are ideal in the graph-based setting since they can operate directly on the similarity matrix and do not require $\boldsymbol{X}$-data support (Chapelle et al., 2006b; Lafferty and Wasserman, 2007). On the other hand, methods of prediction for observations without labels are arguably more complicated and less understood than those from classical supervised settings. A vertex corresponding to an observation without a label provides connections through it which are meaningful to the data structure, and unlabeled data increase

performance if used during training (Culp et al., 2009). The need to extend locally smooth functions into this graph-based setting is an important problem (Chapelle et al., 2006b; Abney, 2008). Applications of graph-based learning include text classification (McCallum et al., 2000), protein interaction (Yamanishi et al., 2004; Kui et al., 2002), chemogenomics in pharmaceuticals (Bredel and Jacoby, 2004), biology and chemistry networks (Lundblad, 2004; Culp et al., 2009), and web data/email (Koprinska et al., 2007). There are also applications where the edges of the graph were constructed using a similarity function generated from feature data (Carreira-Perpiñán and Zemel, 2005; Chapelle et al., 2006b; Jebara et al., 2009).

Harmonic functions provide a natural solution to the problem of extending local classifiers into semi-supervised learning. The definition of a harmonic function depends on two key terms, that is, the boundary (observed labels) and the interior (unlabeled). The boundary choice defines the harmonic function. With a given function estimate on the boundary, the harmonic solution achieves an equilibrium on the interior. Each interior case is an average of its and its neighbors' estimates, so an estimate for an interior observation does not change if averaged a second time. Currently, the authors are aware of only one harmonic approach in the semi-supervised literature. This estimator, referred to as the *clamped harmonic estimator*, sets the boundary equal to its observed labeling. The clamped harmonic estimator in semi-supervised learning was studied and applied to energy optimization (Chapelle et al., 2006b; Abney, 2008), graph-based smoothing (Culp et al., 2009), Gaussian processes (Zhu, 2008), iterative algorithms with large data (Subramanya and Bilmes, 2011), stability methods for transductive learning (Cortes et al., 2008), and other areas (Zhu and Goldberg, 2009).

The clamped harmonic estimator has known shortcomings. First, its performance degradation due to sensitivity to noise in either the support or labeling is well-known. Also, there is no way to estimate a residual, which renders the smoothing technique impossible to use for any inferential analysis, outlier detection, or descriptive analysis. Recent work suggests that the clamped harmonic solution also suffers in circumstances where the size of the boundary is much smaller than that of the interior. The main argument is that the harmonic solution converges to the zero function with spikes within the boundary as the size of the interior grows (Nadler et al., 2009; von Luxburg et al., 2010).

Applications where semi-supervised learning has solid performance as well as an abstraction of such applications into a set of mathematical assumptions is of recent interest (Lafferty and Wasserman, 2007; Azizyan et al., 2013). It is fairly well understood in semi-supervised learning that if two points x_1, x_2 are close in the intrinsic geometry of the probability distribution of X then learning can occur if the conditional probability distributions of $y \mid x_1$ and $y \mid x_2$ are similar. Such a characterization is commonly assumed in semi-supervised learning and often referred to as the *cluster assumption* (Chapelle et al., 2006b). Optimization problems involving minimax error bounds under the cluster and other similar smoothness assumptions is of recent interest (Rigollet, 2007; Lafferty and Wasserman, 2007; Singh et al., 2008). Lafferty and Wasserman (2007) further note the importance of separating semi-supervised smoothness assumptions from other seemingly similar assumptions in manifold learning (Hein et al., 2005; Aswani et al., 2010). The clamped harmonic estimator has been empirically validated to satisfy the cluster assumption, but this, to our knowledge, has not been established rigorously. A key contribution of this work is a condition on the boundary for when any harmonic function is guaranteed to satisfy the cluster assumption.

How semi-supervised approaches compare to supervised alternatives is a looming and important question. In the case of harmonic functions, we are primarily interested in articulating how these

approaches compare to supervised local smoothing classifiers. A significant contribution of this work is extensive analysis and development of harmonic functions in this capacity. In this regard, we show that any harmonic function, no matter how the boundary estimator is generated, can be decomposed as the reweighted average of soft local supervised estimates consisting only of unlabeled predictions. Specifically, the estimate for an interior observation is a weighted average of all the interior local supervised estimates. This work further establishes that interior observations nearest to the boundary carry the weight in the prediction of interior cases.

Harmonic functions and supervised local estimators each use two types of information that describe relationships between the boundary states (labeled) and the interior states (unlabeled). The first type, which we term *labeled adjacent*, involves direct kernel weighted distances from an unlabeled observation to each labeled observation/case. Local supervised approaches essentially form a weighted average of this labeled adjacent information even when an unlabeled case has small adjacency to each labeled case. The second type of information, which we term *labeled connective*, exploits interconnectivity within unlabeled cases to find other unlabeled cases that have stronger adjacency to labeled cases. Harmonic functions propagate the local supervised estimates from unlabeled cases with strong adjacency to some labeled cases to the other unlabeled cases. In short, harmonic functions in semi-supervised learning are purely labeled connective, while local supervised approaches are purely labeled adjacent.

Another key contribution of this work is a new harmonic function approach based off of a joint optimization criterion. The novel use of the joint optimization criterion allows for regularization within semi-supervised learning. Settings of a single regularization parameter can reproduce the extremes, that is, a labeled connective harmonic function estimator or the labeled adjacent soft local supervised estimator, but can also be tuned to any one of a continuum of semi-supervised estimators to compromise between the extremes. It is the only estimator to our knowledge that has been shown to balance between supervised learning and semi-supervised learning in this manner. The benefits of regularization in joint harmonic estimation are empirically assessed with strong results.

The paper is organized as follows. After a brief description of notational conventions in Section 2, the problem is formulated in Section 3. Care is taken to succinctly describe semi-supervised block matrix results in terms of their supervised counterparts, so the stage is set for our main contributions. General results on harmonic functions with regard to the cluster assumption and supervised learning are in Section 4. Section 5 includes the definition of the new regularized joint harmonic function approach and characterization of its mathematical properties. Sections 6 and 7 include empirical tests of the new approach. Section 8 has concluding remarks, and a proof of each Lemma, Proposition, and Theorem is in Appendix A.

2. Notational Conventions

It is common to let $\boldsymbol{A}_{ij}$ represent the entry of a matrix $\boldsymbol{A}$ in row i and column j. A generalization of this $\boldsymbol{A}_{ij}$ notation that is particularly useful in semi-supervised learning is to replace i and j with a list of rows and columns to represent the corresponding sub matrix, so if matrix $\boldsymbol{A}$ is $n \times n$ and sets $L = \{1,2,\ldots,l\}$ and $U = \{l+1,l+2,\ldots,n\}$, then

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{LL} & \boldsymbol{A}_{LU} \\ \boldsymbol{A}_{UL} & \boldsymbol{A}_{UU} \end{pmatrix}. \tag{1}$$

The usefulness of Partitioning (1) will become clear when attention turns back to discussion of the sets of labeled L and unlabeled U cases in the semi-supervised learning context of Section 3. Denote

$$\boldsymbol{A}^*_{LL} = \boldsymbol{A}_{LL} - \boldsymbol{A}_{LU}\boldsymbol{A}^{-1}_{UU}\boldsymbol{A}_{UL} \quad (\boldsymbol{A}_{UU} \text{ Block Schur Complement of } \boldsymbol{A}). \tag{2}$$

Note the important distinction between $\boldsymbol{A}_{LL}$ in Display (1) and $\boldsymbol{A}^*_{LL}$ in Display (2). Schur complements and some of their most basic properties given in Remark 1 play a key role in the methods to come as well as in the Appendix A proofs. Table 1 summarizes all of our matrix algebra conventions for future reference.

Notation	Definition
$\mathcal{N}(\boldsymbol{A})$	Null space of matrix $\boldsymbol{A}$.
$\boldsymbol{A} \geq 0$	Matrix $\boldsymbol{A}$ with all nonnegative entries ($>$ for positive).
$\boldsymbol{A} \succeq 0$	Positive semi-definite symmetric matrix $\boldsymbol{A}$ ($\succ$ for positive definite).
$\rho^{(i)}(\boldsymbol{A})$	ith largest modulus of the eigenvalues of a square matrix $\boldsymbol{A}$.
$\rho(\boldsymbol{A})$	Spectral radius of a square matrix $\boldsymbol{A}$, that is, $\rho(\boldsymbol{A}) = \rho^{(1)}(\boldsymbol{A})$.
$\boldsymbol{A}_{LL}$	Upper-left sub matrix in Partitioning (1) of a square matrix $\boldsymbol{A}$.
$\boldsymbol{A}^*_{LL}$	$\boldsymbol{A}_{UU}$ Block Schur Complement (2) of matrix $\boldsymbol{A}$ with Partitioning (1).

Table 1: List of notational conventions.

Remark 1 *Based on the Partitioning (1), it is well known that if $\boldsymbol{A}_{UU}$ is invertible then $\boldsymbol{A}$ is invertible if and only if $\boldsymbol{A}^*_{LL}$ is invertible. In the case that $\boldsymbol{A} \succeq 0$ (i.e., $\boldsymbol{A}$ is symmetric and positive semi-definite), this result becomes if $\boldsymbol{A}_{UU} \succ 0$ then $\boldsymbol{A} \succ 0$ if and only if $\boldsymbol{A}^*_{LL} \succ 0$.*

3. Problem Set-Up

In graph-based semi-supervised learning, partially labeled data are in the form of a weighted graph. Vertices $\{1,\cdots,n\}$ represent the n observations, and edges the values of a correspondence between each pair of observations. The $n \times n$ symmetric matrix $\boldsymbol{W}$ with $\boldsymbol{W}_{ij} \geq 0$ is the adjacency matrix of the *weighted graph* $(\{1,\cdots,n\}, \boldsymbol{W})$ or graph $\boldsymbol{W}$ for brevity. For this particular weighted graph, additionally assume $\boldsymbol{W}_{ij} \leq 1$ and $\boldsymbol{W}_{ii} = 1$. In some applications, $\boldsymbol{W}$ must be constructed from an $n \times p$ data matrix $\boldsymbol{X}$, for example,

$$\boldsymbol{W}_{ij} = K_\lambda(x_i, x_j),$$

where kernel function $K_\lambda(x_i, x_j)$ is applied to each pair of rows of $\boldsymbol{X}$ to form $\boldsymbol{W}$. Experimental Sections 6 and 7 include examples of each type, that is, $\boldsymbol{W}$ observed directly and $\boldsymbol{W}$ generated from $\boldsymbol{X}$. For now, simply assume that the symmetric matrix $\boldsymbol{W}$ is in hand.

The *training response* is

$$Y(Y_U) = \begin{pmatrix} Y_L \\ Y_U \end{pmatrix} \in \mathbb{R}^n, \text{ where } Y_U \in \mathbb{R}^{|U|}, \tag{3}$$

and the data partition into two observed subsets $\{1,\cdots,n\} = L \cup U$. Subset L is the set of all *boundary states*, whereas U is that for *interior states*. The subsets are distinguished by the labeling

function. The boundary states have an observed labeling vector Y_L, while the labelings for the interior states go unobserved. We assert the missing at random assumption and assume that L was initially a random subset of $\{1,\cdots,n\}$, but for ease of notation, the data were subsequently sorted so that boundary observations are first in the indexing. The vector of latent variables Y_U is comprised of the unknown labelings for the interior. Our joint optimization based method defined later in Section 5 involves the training response. The solution to this joint optimization problem provides the capacity for transductive or semi-supervised learning as will be illustrated later in Section 7.

Next, general graph theory results are discussed and applied to graph $\boldsymbol{W}$. In particular, Laplacian and stochastic smoother matrices corresponding to graph $\boldsymbol{W}$ are defined, and the relationships between these three matrices are discussed briefly. It is fundamental to think about the general idea being applied to graph $\boldsymbol{W}$ because later they will be applied to a particular graph with vertex set L in each of the Sections 3.1-3.3. These three graphs on L to be introduced in Sections 3.1-3.3 help one understand a semi-supervised technique through a decomposition of L to L connectivities in the larger graph $\boldsymbol{W}$ on $L \cup U$.

The *Laplacian* of $\boldsymbol{W}$ is $\boldsymbol{\Delta} = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{D} = \text{diag}(\boldsymbol{W}\vec{1})$ is the *degree matrix* of $\boldsymbol{W}$. Proposition 7 is a well-known result on $\boldsymbol{\Delta}$ (Belkin et al., 2006).

Proposition 7 *Laplacian* $\boldsymbol{\Delta} \succeq 0$.

The square matrix $\boldsymbol{S} = \boldsymbol{D}^{-1}\boldsymbol{W}$ is a *stochastic smoother*, that is, $\boldsymbol{S} \geq 0$ and $\boldsymbol{S}\vec{1} = \vec{1}$, so 1 is an eigenvalue of $\boldsymbol{S}$. Proposition 8 further establishes that $\rho(\boldsymbol{S}) = 1$.

Proposition 8 *If* $\boldsymbol{W} \succeq 0$ *then each eigenvalue of* $\boldsymbol{S} = \boldsymbol{D}^{-1}\boldsymbol{W}$ *is an element of* $[0,1]$.

The identity $\boldsymbol{\Delta} = \boldsymbol{D}(\boldsymbol{I} - \boldsymbol{S})$ helps demonstrate that

$$\boldsymbol{\Delta}\nu = \vec{0} \iff \boldsymbol{S}\nu = \nu, \tag{4}$$

that is, $\mathcal{N}(\boldsymbol{\Delta})$ equals the eigenspace of $\boldsymbol{S}$ corresponding to eigenvalue 1. An eigenvalue decomposition of $\boldsymbol{\Delta}$ or $\boldsymbol{S}$ provides a way to compute the number of connected components in graph $\boldsymbol{W}$. One simply counts the multiplicity of eigenvalue 0 for $\boldsymbol{\Delta}$ by Remark 2 or equivalently eigenvalue 1 for $\boldsymbol{S}$ by Display (4).

The graphs in Sections 3.1-3.3 are based on partitioning the adjacency, stochastic smoother, and Laplacian matrices of graph $\boldsymbol{W}$ by L and U. Using Section 2 notation and Display (1) in particular, this is

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{W}_{LL} & \boldsymbol{W}_{LU} \\ \boldsymbol{W}_{UL} & \boldsymbol{W}_{UU} \end{pmatrix}, \quad \boldsymbol{S} = \begin{pmatrix} \boldsymbol{S}_{LL} & \boldsymbol{S}_{LU} \\ \boldsymbol{S}_{UL} & \boldsymbol{S}_{UU} \end{pmatrix}, \quad \boldsymbol{\Delta} = \begin{pmatrix} \boldsymbol{\Delta}_{LL} & \boldsymbol{\Delta}_{LU} \\ \boldsymbol{\Delta}_{UL} & \boldsymbol{\Delta}_{UU} \end{pmatrix}. \tag{5}$$

The entries of $\boldsymbol{W}_{LL}$ and $\boldsymbol{W}_{UU}$ are similarities within the boundary and interior, respectively, while $\boldsymbol{W}_{LU} = \boldsymbol{W}_{UL}^T$ contain the similarities between boundary and interior observations. Analogous interpretations extend to the other matrices partitioned in Display (5). For the diagonal degree matrix $\boldsymbol{D} \geq 0$, define the $|L| \times |L|$ diagonal matrices $\widetilde{\boldsymbol{D}}_{LL} = \text{diag}(\boldsymbol{W}_{LL}\vec{1}) \geq 0$ and $\widetilde{\boldsymbol{D}}_{LU} = \text{diag}(\boldsymbol{W}_{LU}\vec{1}) \geq 0$ and the $|U| \times |U|$ diagonal matrices $\widetilde{\boldsymbol{D}}_{UU} = \text{diag}(\boldsymbol{W}_{UU}\vec{1}) \geq 0$ and $\widetilde{\boldsymbol{D}}_{UL} = \text{diag}(\boldsymbol{W}_{UL}\vec{1}) \geq 0$, so that

$$\boldsymbol{D} = \begin{pmatrix} \boldsymbol{D}_{LL} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_{UU} \end{pmatrix} = \begin{pmatrix} \widetilde{\boldsymbol{D}}_{LL} + \widetilde{\boldsymbol{D}}_{LU} & \boldsymbol{0} \\ \boldsymbol{0} & \widetilde{\boldsymbol{D}}_{UL} + \widetilde{\boldsymbol{D}}_{UU} \end{pmatrix}.$$

Next, supervised, offset, and semi-supervised weighted graphs are studied in Sections 3.1-3.3 to assist in a deep understanding of a semi-supervised boundary estimation method.

Remark 2 *Vertices i and j are adjacent in graph $\boldsymbol{W}$ if $\boldsymbol{W}_{ij} > 0$, and are connected if there exists a sequence of vertices starting with i and ending with j such that consecutive vertices throughout the sequence are adjacent. The concept of connectedness partitions the vertices into some number of connected components, and each vertex in a connected component is connected to any other vertex in that component. Basic structure of a weighted graph includes the number of connected components and whether or not any given pair of vertices is in the same connected component. Both of these properties are encoded in particular eigenvectors of the graph's Laplacian matrix and stochastic smoother. Just take the binary vector in $\mathbb{R}^n$ that indicates observations in a connected component of $\boldsymbol{W}$. The set of all such binary vectors over all connected components is an orthogonal basis for $\mathcal{N}(\boldsymbol{\Delta})$, so the dimension of $\mathcal{N}(\boldsymbol{\Delta})$ equals the number of connected components. Furthermore, it is obvious that the vectors in this basis sum to $\vec{1} \in \mathcal{N}(\boldsymbol{\Delta})$.*

3.1 The Supervised Case

The supervised local kernel smoother at any point x_i is

$$\widetilde{f}(i) = \frac{\sum_{j\in L} K_\lambda(x_i,x_j)y_j}{\sum_{j\in L} K_\lambda(x_i,x_j)} \approx E[Y_i \mid X_i = x_i]$$

and is often called a Nadaraya-Watson kernel regression estimator (Hastie et al., 2001, Chapter 6). When applied to $L \cup U$, this estimator is

$$\widetilde{f} = \begin{pmatrix} \widetilde{f}_L \\ \widetilde{f}_U \end{pmatrix} = \begin{pmatrix} \widetilde{\boldsymbol{S}}_{LL} \\ \widetilde{\boldsymbol{S}}_{UL} \end{pmatrix} Y_L = \begin{pmatrix} \widetilde{\boldsymbol{D}}_{LL}^{-1}\boldsymbol{D}_{LL}\boldsymbol{S}_{LL} \\ \widetilde{\boldsymbol{D}}_{UL}^{-1}\boldsymbol{D}_{UU}\boldsymbol{S}_{UL} \end{pmatrix} Y_L, \tag{6}$$

where $\widetilde{\boldsymbol{S}}_{LL} = \widetilde{\boldsymbol{D}}_{LL}^{-1}\boldsymbol{W}_{LL}$ and $\widetilde{\boldsymbol{S}}_{UL} = \widetilde{\boldsymbol{D}}_{UL}^{-1}\boldsymbol{W}_{UL}$.

The supervised boundary estimator $\widetilde{f}_L = \widetilde{\boldsymbol{S}}_{LL}Y_L$ in Display (6) is based on the *supervised graph* $(L, \boldsymbol{W}_{LL})$. The supervised graph is the subgraph of $\boldsymbol{W}$ on L and has

$$\begin{array}{rcll} \widetilde{\boldsymbol{\Delta}}_{LL} & = & \widetilde{\boldsymbol{D}}_{LL} - \boldsymbol{W}_{LL} = \boldsymbol{\Delta}_{LL} - \boldsymbol{D}_{LL} & \text{(Supervised Laplacian)}, \\ \widetilde{\boldsymbol{S}}_{LL} & = & \widetilde{\boldsymbol{D}}_{LL}^{-1}\boldsymbol{W}_{LL} & \text{(Supervised Stochastic Smoother)}. \end{array}$$

The supervised smoothed value $\widetilde{f}_i$ for $i \in L$ is the probability weighted average of Y_L with weights from the ith row of $\widetilde{\boldsymbol{S}}_{LL}$, so $\widetilde{f}_i$ is based on relative strength of adjacencies within L, which might be depicted by $L \to L$. The supervised graph incorporates neither non-adjacent vertices nor U. Estimator $\widetilde{f}_L$ is also the solution to

$$\min_{f_L}(Y_L - f_L)^T\boldsymbol{W}_{LL}(Y_L - f_L) + f_L^T\widetilde{\boldsymbol{\Delta}}_{LL}f_L.$$

Supervised predictions of the interior from Display (6) are $\widetilde{f}_U = \widetilde{\boldsymbol{S}}_{UL}Y_L$. If $\widetilde{\boldsymbol{D}}_{UL_{ii}} = 0$ for some $i \in U$ then this supervised estimator is not defined for interior observation i, so this estimator exists for all $i \in U$ if and only if $\widetilde{\boldsymbol{D}}_{UL} \succ 0$, that is,

$$\nu^T\widetilde{\boldsymbol{D}}_{UL}\nu > 0 \text{ for any non-zero } \nu \in \mathbb{R}^{|U|}. \tag{7}$$

Condition (7) holds if and only if each unlabeled observation is adjacent to a labeled observation. This adjacency condition is a stringent requirement, especially when the proportion of labeled observations $|L|/n$ is small, and one might correctly guess that such a rigid requirement is not necessary if a semi-supervised harmonic function approach from Section 4 is taken.

3.2 The Offset Case

In this section, three $|L| \times |L|$ matrices $\boldsymbol{W}_{LUL}$, $\boldsymbol{\Delta}_{LUL}$, and $\boldsymbol{S}_{LUL}$ are defined, and it is shown that they correspond to the adjacency, Laplacian, and stochastic smoother matrices of a weighted graph on vertex set L, which we call the *offset graph*. These matrices are

$$\begin{array}{rcll} \boldsymbol{W}_{LUL} & = & \boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL} & \text{(Offset Graph with Vertex Set } L\text{)}, \\ \boldsymbol{\Delta}_{LUL} & = & \widetilde{\boldsymbol{D}}_{LU} - \boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL} & \text{(Offset Laplacian)}, \\ \boldsymbol{S}_{LUL} & = & \widetilde{\boldsymbol{D}}_{LU}^{-1}\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL} & \text{(Offset Stochastic Smoother)}. \end{array}$$

Recall the necessary and sufficient adjacency condition in Display (7) for the uniqueness of the supervised estimator for all n observations. An intuitive condition for the uniqueness of a semi-supervised estimator for all n observations is that each connected component of $\boldsymbol{W}$ includes an observation from L, that is,

$$\nu^T \widetilde{\boldsymbol{D}}_{UL}\nu > 0 \text{ for any non-zero } \nu \in \mathcal{N}\left(\widetilde{\boldsymbol{D}}_{UU} - \boldsymbol{W}_{UU}\right). \tag{8}$$

Apply Remark 2 to subgraph $(U, \boldsymbol{W}_{UU})$ to justify this practical interpretation of Condition (8). The connectedness to L condition in Display (8) is less restrictive than the adjacency to L condition in Display (7), and Condition (8) implies that $\boldsymbol{W}$ has at most $|L|$ connected components. Proposition 10 establishes that Condition (8) is equivalent to the existence of $\boldsymbol{\Delta}_{UU}^{-1}$, a matrix involved in the definition of the offset graph.

Proposition 10 *If $\boldsymbol{W} \succeq 0$ then the following conditions are equivalent.*

(a) $\boldsymbol{\Delta}_{UU} \succ 0$.

(b) $\rho(\boldsymbol{S}_{UU}) < 1$.

(c) $\nu^T \widetilde{\boldsymbol{D}}_{UL}\nu > 0$ *for any non-zero* $\nu \in \mathcal{N}(\widetilde{\boldsymbol{D}}_{UU} - \boldsymbol{W}_{UU})$.

Condition (b) from Proposition 10 guarantees the convergence of the geometric matrix series with terms $\boldsymbol{S}_{UU}^{\ell} = \boldsymbol{\mathcal{O}}\boldsymbol{\mathcal{D}}^{\ell}\boldsymbol{\mathcal{O}}^{-1}$, where $\boldsymbol{\mathcal{O}}\boldsymbol{\mathcal{D}}\boldsymbol{\mathcal{O}}^{-1}$ is the eigendecomposition of $\boldsymbol{S}_{UU}$, so

$$\boldsymbol{D}_{LL}^{-1}\boldsymbol{W}_{LUL} = \boldsymbol{D}_{LL}^{-1}\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL} = \boldsymbol{S}_{LU}\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL} = \sum_{\ell=0}^{\infty} \boldsymbol{S}_{LU}\boldsymbol{S}_{UU}^{\ell}\boldsymbol{S}_{UL} \geq 0, \tag{9}$$

where the inequality holds because $\boldsymbol{S}_{LU}\boldsymbol{S}_{UU}^{\ell}\boldsymbol{S}_{UL} \geq 0$ for each $\ell = 0, 1, \ldots$. Thus, $\boldsymbol{W}_{LUL} \geq 0$ is a valid weighted graph on L, since it's symmetric by definition. By the Laplacian property $\boldsymbol{\Delta}\vec{1} = \vec{0}$ and Partitioning (5), $\boldsymbol{\Delta}_{UL}\vec{1} = -\boldsymbol{\Delta}_{UU}\vec{1}$ and $\boldsymbol{\Delta}_{LU}\vec{1} = -\widetilde{\boldsymbol{D}}_{LU}\vec{1}$, so the degree matrix of $\boldsymbol{W}_{LUL}$ is $\text{diag}(\boldsymbol{W}_{LUL}\vec{1}) = \widetilde{\boldsymbol{D}}_{LU}$. Thus, the Laplacian and stochastic smoother of offset graph $\boldsymbol{W}_{LUL}$ are also established as matrices $\boldsymbol{\Delta}_{LUL}$ and $\boldsymbol{S}_{LUL}$ defined earlier.

The geometric matrix series in Display (9) provides a clear interpretation of each adjacency in offset graph $\boldsymbol{W}_{LUL}$. A pair of labeled observations is adjacent in $\boldsymbol{W}_{LUL}$ if and only if they are connected in $\boldsymbol{W}$ through a sequence of unlabeled observations; this type of connectedness might be depicted by $L \to U \leftrightarrow U \to L$. The offset boundary estimator is $(\boldsymbol{S}_{LUL}Y_L)_i$ for $i \in L$, that is, the probability weighted average of Y_L with weights from the ith row of $\boldsymbol{S}_{LUL}$. The probability weight on Y_{L_j} for $j \in L$ is $\boldsymbol{S}_{LUL_{ij}}$, and this weight will be relatively large if i has "strong" adjacencies to vertices in a "strongly adjacent" U network that is "strongly adjacent" to j. These are the only types of connectivity that matter in the offset case. For example, the adjacency between i and j simply does not factor into the offset based estimator.

3.3 The Semi-Supervised Case

The semi-supervised adjacency matrix is simply the sum of those from the supervised and offset cases, that is,

$$\boldsymbol{W}_{LL}+\boldsymbol{W}_{LUL} \quad \text{(Semi-Supervised Graph with Vertex Set } L\text{)}.$$

The semi-supervised Laplacian is thus the sum of positive semi-definite Laplacians

$$\begin{aligned}\boldsymbol{\Delta}^{\star}_{LL} &= \overbrace{\widetilde{\boldsymbol{D}}_{LL}-\boldsymbol{W}_{LL}}^{\substack{\text{Supervised}\\ \text{Laplacian}}}+\overbrace{\widetilde{\boldsymbol{D}}_{LU}-\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL}}^{\text{Offset Laplacian}} && \text{(Semi-Supervised Laplacian)} \\ &= \boldsymbol{\Delta}_{LL}-\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL} && (\boldsymbol{\Delta}_{UU} \text{ Block Schur Complement of } \boldsymbol{\Delta}).\end{aligned} \tag{10}$$

Refer to Section 2 and Display (2) for Schur complements.

The semi-supervised stochastic smoother is $\boldsymbol{M}_{LL}=\boldsymbol{D}_{LL}^{-1}(\boldsymbol{W}_{LL}+\boldsymbol{W}_{LUL})$. For more insight, first define the diagonal matrix $\boldsymbol{Q}_L=\boldsymbol{D}_{LL}^{-1}\widetilde{\boldsymbol{D}}_{LL}\succ 0$, which stores the proportion of each case's total similarities over all cases $L\cup U$ that is within L, that is,

$$\boldsymbol{Q}_{L_{ii}}=\frac{\sum_{j\in L}\boldsymbol{W}_{ij}}{\sum_{j\in L\cup U}\boldsymbol{W}_{ij}}.$$

Matrix $\boldsymbol{Q}_L$ provides the case-by-case probability weighted average compromise between the supervised and offset stochastic smoothers that is the semi-supervised stochastic smoother

$$\boldsymbol{M}_{LL}=\boldsymbol{Q}_L\widetilde{\boldsymbol{S}}_{LL}+(\boldsymbol{I}-\boldsymbol{Q}_L)\boldsymbol{S}_{LUL} \quad \text{(Semi-Supervised Stochastic Smoother)}.$$

More factorization produces yet another equivalent form

$$\boldsymbol{M}_{LL}=\boldsymbol{S}_{LL}+\boldsymbol{S}_{LU}(\boldsymbol{I}-\boldsymbol{S}_{UU})^{-1}\boldsymbol{S}_{UL} \quad (\boldsymbol{S}_{UU} \text{ Stochastic Complement of } \boldsymbol{S}). \tag{11}$$

Adjacencies accumulate in semi-supervised graph $\boldsymbol{W}_{LL}+\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL}$ due to exactly two types of connectedness among the labeled observations in graph $\boldsymbol{W}$: (i) supervised $L\rightarrow L$ and (ii) offset $L\rightarrow U\leftrightarrow U\rightarrow L$. The prediction for a case $i\in L$ puts more weight on the supervised prediction for large $\boldsymbol{Q}_{L_{ii}}$ and on the offset prediction for large $1-\boldsymbol{Q}_{L_{ii}}$, so $\boldsymbol{M}_{LL}$ is always a practical probability weighted average of the estimators based on graphs $\boldsymbol{W}_{LL}$ and $\boldsymbol{W}_{LUL}$. The connectedness of labeled vertices in the semi-supervised graph is the same as that in the full graph $\boldsymbol{W}$, but types of connectedness outside (i) and (ii) don't get incorporated into semi-supervised predictions (see Remark 3).

The decomposition of the semi-supervised graph into supervised and offset graphs is displayed concisely in Figure 1. While it is not too hard to compute the Laplacian or stochastic smoother from the weighted graph, no other offset or semi-supervised representation can be fully recovered from just the Laplacian or just the smoother. However, it is possible to recover $\boldsymbol{W}$ from $\boldsymbol{S}$ because $\boldsymbol{W}_{ii}=1$ is known.

Additional insight into the inter-workings of the semi-supervised smoother is gleaned through analytical eigenvalue results. First, $\boldsymbol{\Delta}^{\star}_{LL}=\boldsymbol{D}_{LL}(\boldsymbol{I}-\boldsymbol{M}_{LL})$, so

$$\boldsymbol{\Delta}^{\star}_{LL}\nu=\vec{0} \iff \boldsymbol{M}_{LL}\nu=\nu$$

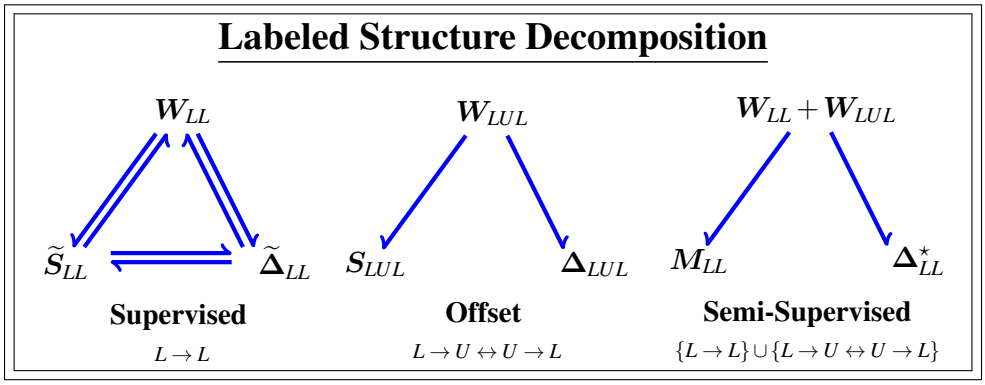


Figure 1: Matrix representations of weighted graphs each with vertex set L: adjacency (top), stochastic smoother (bottom left), and Laplacian (bottom right). Each semi-supervised labeled representation is a linear combination of the corresponding supervised and offset representations. Harpoons indicate that the representation after the barb can be computed from that on the other end.

provides a second example of the general relationship between a smoother and its Laplacian (reference Display (4) for that between $\boldsymbol{\Delta}$ and $\boldsymbol{S}$). To analytically break down $\mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL})$ (and hence the eigenspace of $\boldsymbol{M}_{LL}$ corresponding to eigenvalue 1), first recall the decomposition of Laplacian $\boldsymbol{\Delta}^{\star}_{LL}$ in Equation (10) as the sum of positive semi-definite Laplacians. Thus,

$$\mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL}) = \mathcal{N}\left(\widetilde{\boldsymbol{\Delta}}_{LL}\right) \cap \mathcal{N}(\boldsymbol{\Delta}_{LUL}) \subseteq I\!R^{|L|}.$$

Certainly $\vec{1} \in \mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL})$, and a particular orthogonal basis of binary vectors for $\mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL})$ is given by Remark 2. Each basis vector indicates vertices in a connected component of the semi-supervised graph, and so they partition L and sum to $\vec{1}$. Similarly, partitions of L corresponding to the connected components of the supervised and offset graphs correspond to orthogonal bases of binary vectors for $\mathcal{N}(\widetilde{\boldsymbol{\Delta}}_{LL})$ and $\mathcal{N}(\boldsymbol{\Delta}_{LUL})$. The operation of intersecting $\mathcal{N}(\widetilde{\boldsymbol{\Delta}}_{LL})$ and $\mathcal{N}(\boldsymbol{\Delta}_{LUL})$ can never increase the dimension of the resulting $\mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL})$ and is equivalent to increasing connectivity by producing the coarsest possible partition of L that can be made by both partitions (of L corresponding to $\mathcal{N}(\widetilde{\boldsymbol{\Delta}}_{LL})$ and $\mathcal{N}(\boldsymbol{\Delta}_{LUL})$) via unions of their respective subsets.

Supervised graph $\boldsymbol{W}_{LL}$ is a subgraph of $\boldsymbol{W}$. They have the same adjacencies in L, but $\boldsymbol{W}_{LL}$ can only reduce connectivity in L relative to that in $\boldsymbol{W}$. The addition of the offset $\boldsymbol{W}_{LUL}$ to $\boldsymbol{W}_{LL}$ achieves the same level of connectedness in L as $\boldsymbol{W}$, but more importantly introduces offset adjacencies in the semi-supervised graph not found in the supervised graph. It is the adjacencies in the semi-supervised graph that determine non-zero smoother weights (see Remark 3). In spite of this, the connectedness structure of the semi-supervised graph is still important so that one understands the smoother properties via its eigenvalue decomposition. If a condition from Proposition 10 holds, then each connected component of $\boldsymbol{W}$ includes a vertex from L. In this case, the dimension of $\mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL}) \subseteq I\!R^{|L|}$ equals the dimension of $\mathcal{N}(\boldsymbol{\Delta}) \subseteq I\!R^{|L \cup U|}$. Intuitively, we view $\boldsymbol{M}_{LL}$ as a labeled stochastic smoother with respect to the observed response Y_L, while $\boldsymbol{S}$ is a stochastic smoother with respect to the training response $Y(Y_U)$.

Remark 3 *Semi-supervised graph $\boldsymbol{W}_{LL}+\boldsymbol{W}_{LUL}$ on L keeps the meaningful connectedness structure of the full graph $\boldsymbol{W}$ on $L\cup U$. A pair of labeled observations are in the same connected component of one of these graphs if and only if the same is true in the other graph. This follows because adjacent boundary vertices in $\boldsymbol{W}_{LL}+\boldsymbol{W}_{LUL}$ are connected in $\boldsymbol{W}$ via either a sequence of labeled vertices (supervised) or a sequence of unlabeled vertices (offset), and sequences of these two types of connectivities in $\boldsymbol{W}$ can build any type connectivity that exists in $\boldsymbol{W}$ from an $i\in L$ to $j\in L$. It follows that*

$$\nu=\begin{pmatrix}\nu_L\\ \nu_U\end{pmatrix}\in\mathcal{N}(\boldsymbol{\Delta})\subseteq\mathbb{R}^{|L\cup U|}\implies\nu_L\in\mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL})\subseteq\mathbb{R}^{|L|} \tag{12}$$

(refer to Remark 2).

Let $i\in L$ and $j\in L$. Probability weight $\boldsymbol{M}_{LL_{ij}}$ is that for Y_{L_j} in the semi-supervised smoothed value for Y_{L_i}. It should come as no surprise that a sufficient condition for $\boldsymbol{M}_{LL_{ij}}=0$ is that boundary vertices i and j are not in the same connect component of $\boldsymbol{W}$, but this condition is not necessary. The necessary and sufficient condition for $\boldsymbol{M}_{LL_{ij}}>0$ is that i and j are adjacent in at least one graph $\boldsymbol{W}_{LL}$ or $\boldsymbol{W}_{LUL}$. The hypothetical situation where i and j are in the same connect component of $\boldsymbol{W}$ and $\boldsymbol{M}_{LL_{ij}}=0$ is possible if boundary vertices i and j are connected in the full graph $\boldsymbol{W}$ but not through a pure sequence of all boundary (or of all interior) vertices.

4. Harmonic Functions in Semi-Supervised Learning

Harmonic functions form the basis for the connection between electrical networks and random walks (Doyle and Snell, 1984). The use of harmonic estimation in semi-supervised learning is discussed extensively in its relation to random walks, electrical networks, and energy optimization (Zhu et al., 2003).

A function $h:\mathcal{V}\to\mathbb{R}$ is *harmonic* with respect to a stochastic matrix $\boldsymbol{S}$ if

$$f_i=\sum_{\ell\in L\cup U}\boldsymbol{S}_{i\ell}f_\ell\quad\text{for each } i\in U, \tag{13}$$

where $f_i=h(i)$ (Zhu et al., 2003; Abney, 2008). In matrix form, the implication of Equation (13) on a resulting *harmonic estimator* $f\in\mathbb{R}^n$ is

$$\boldsymbol{S}f=\begin{pmatrix}\boldsymbol{S}_{LL}f_L+\boldsymbol{S}_{LU}f_U\\ \boldsymbol{S}_{UL}f_L+\boldsymbol{S}_{UU}f_U\end{pmatrix}=\begin{pmatrix}(\boldsymbol{S}f)_L\\ f_U\end{pmatrix}. \tag{14}$$

In the case of a harmonic estimator in Display (14), it follows by Display (12) that $(\boldsymbol{S}f)_L=f_L$ if and only if $f_L\in\mathcal{N}(\boldsymbol{\Delta}^{\star}_{LL})$. In other words, $\boldsymbol{S}f=f$ holds for a harmonic estimator f if and only if f_L is constant within the connected components of $\boldsymbol{W}$. This precise concept of when $\boldsymbol{S}f=f$ is in tandem with the practical application of a judiciously chosen harmonic estimator under the cluster assumption studied further in Section 4.1.

A question not addressed in the above discussion is the existence and uniqueness of a harmonic estimator f. This mathematical matter is solved in two cases $\rho(\boldsymbol{S}_{UU})<1$ and $\rho(\boldsymbol{S}_{UU})=1$, which are collectively exhaustive by Lemma 9 in Appendix A. First, consider the case of $\rho(\boldsymbol{S}_{UU})<1$ (or any other equivalent condition from Proposition 10), so that $(\boldsymbol{I}-\boldsymbol{S}_{UU})^{-1}$ exists. In this case, the unique estimator for the interior $f_U=(\boldsymbol{I}-\boldsymbol{S}_{UU})^{-1}\boldsymbol{S}_{UL}f_L$ is a linear transformation of the boundary

estimate. If one uses this unique solution for the interior as well as the stochastic complement representation of $\boldsymbol{M}_{LL}$ from Equation (11), then Equation (14) simplifies to

$$\boldsymbol{S}f = \begin{pmatrix} (\boldsymbol{S}f)_L \\ f_U \end{pmatrix} = \begin{pmatrix} \boldsymbol{M}_{LL} \\ (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} \end{pmatrix} f_L. \tag{15}$$

The left of Equation (15) is an $n \times n$ times $n \times 1$ matrix multiplication, whereas the right is an $n \times |L|$ times an $|L| \times 1$. Next, the case of $\rho(\boldsymbol{S}_{UU}) = 1$ implies that at least one connected component in $\boldsymbol{W}$ contains all interior observations, that is, Condition (8) does not hold. So with given estimate f_L, a harmonic estimate f_U exists, but is not unique because there is an arbitrary choice for a constant labeling within each pure interior connected component. The assumption $\rho(\boldsymbol{S}_{UU}) < 1$ used throughout most of Sections 3 and 4 avoids this arbitrary nature of harmonic estimators when $\rho(\boldsymbol{S}_{UU}) = 1$. The subtlety in the case of $\rho(\boldsymbol{S}_{UU}) = 1$ is directly overcome by methods of regularization presented later in Section 5.

The maximum principle states that a harmonic solution is bounded above and below by the boundary estimate (Doyle and Snell, 1984). The uniqueness principle, which applies in the case of $\rho(\boldsymbol{S}_{UU}) < 1$, states that if two harmonic functions are applied with the same boundary estimate f_L then they must produce the same interior estimate f_U. One thing that is clear from each of these principles is that a harmonic estimate f_U of the interior is a function of the boundary estimate f_L. While the semi-supervised boundary estimator $f_L = \boldsymbol{M}_{LL} Y_L$ was thoroughly developed in Section 3, the plethora of competing boundary estimators is a focus of Section 4.1.

4.1 The Cluster Assumption and Boundary Estimation

The *cluster assumption* states that observations close in proximity should have similar labels. Our main objective is to understand how this concept relates to classifiers. Let ψ be an arbitrary classifier trained with weighted graph $\boldsymbol{W}$ and arbitrary response Y_L. We say that ψ is a *cluster assumption classifier* if ψ is guaranteed to satisfy

$$\psi \in \mathcal{N}(\boldsymbol{\Delta}) \text{ and } \psi_L = Y_L \iff Y_L \in \mathcal{N}(\boldsymbol{\Delta}^\star_{LL}). \tag{16}$$

Suppose the response is constant within the connected components of $\boldsymbol{W}$. Condition (16) guarantees that a cluster assumption classifier classifies each interior observation with the unique label observed within its connected component (refer to Remarks 2 and 3).

Let f be a harmonic function trained from the weighted graph $\boldsymbol{W}$ and response Y_L. In order for f to also be a cluster assumption classifier, the boundary must be estimated with $f_L = Y_L$ for any $Y_L \in \mathcal{N}(\boldsymbol{\Delta}^\star_{LL})$, that is, $Y_L \in \mathcal{N}(\boldsymbol{\Delta}^\star_{LL}) \Longrightarrow \boldsymbol{S}f = f$ and $f \in \mathcal{N}(\boldsymbol{\Delta})$. Harmonic functions that are cluster assumption classifiers are also useful in circumstances when $\boldsymbol{W}$ has only one connected component. Suppose there are weak adjacencies less than some small $\varepsilon/n > 0$ between clusters, and pairs within clusters are connected by an edge path with adjacencies exceeding ε. Then decomposition $\boldsymbol{W} = \boldsymbol{W}_{\text{weak}} + \boldsymbol{W}_{\text{strong}}$, where $\boldsymbol{W}_{\text{weak}_{ij}} = \min\{\varepsilon/n, \boldsymbol{W}_{ij}\}$, produces connected components in the strong graph that correspond to clusters. The cluster assumption holds on the strong graph. Now, for any $f \in \mathcal{N}(\boldsymbol{\Delta}_{\text{strong}})$, $\boldsymbol{S}f \approx \boldsymbol{S}_{\text{strong}} f \in \mathcal{N}(\boldsymbol{\Delta}_{\text{strong}})$ because the smoother $\boldsymbol{S}$ is a row wise probability weighted average of the strong and weak smoothers that puts a low weight on the weak smoother. If $f_L = Y_L \in \mathcal{N}\left(\boldsymbol{\Delta}^\star_{LL_{\text{strong}}}\right)$ such that $Y_{L_i} = 1$ on a connected component of $\boldsymbol{W}_{\text{strong}}$ and $Y_{L_i} = -1$ elsewhere, then $\text{sign}(\boldsymbol{S}f) \in \mathcal{N}(\boldsymbol{\Delta}_{\text{strong}})$, so the hard labels classify in accordance with the cluster

assumption, which is consistent with the empirical evidence in the literature (Chapelle et al., 2006b; Abney, 2008).

The simplest boundary estimate for a harmonic estimator is the *clamped harmonic estimator* $f_L = Y_L$ (Zhu et al., 2003; Abney, 2008). The clamped harmonic estimator can be motivated as solving

$$\min_{f_L} (Y_L - f_L)^T (Y_L - f_L)$$

to obtain the boundary estimator $f_L = Y_L$ and then enforcing Equation (15) to define a harmonic estimator by setting $f_U = (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} f_L$.

This is not the only possible harmonic estimator because one can use any boundary estimator to develop a harmonic estimator. For example, consider

$$\min_{f} (Y_L - f_L)^T (\boldsymbol{W}_{LL} + \boldsymbol{W}_{LUL})(Y_L - f_L) + f^T \boldsymbol{\Delta} f, \tag{17}$$

where the loss function is based off of the semi-supervised graph developed in Section 3. The solution to Optimization (17) is a harmonic function with the boundary estimate $f_L = \boldsymbol{M}_{LL} Y_L$ from Section 3.3. The reason why Optimization (17) produces a harmonic function can be seen by studying the optimization of a generalized labeled loss function with penalty

$$\min_{f} L(Y_L, f_L) + \eta f^T \boldsymbol{\Delta} f, \tag{18}$$

where $L(Y_L, Y_L) \leq L(Y_L, f_L)$ for any f_L. Since this loss function is independent of f_U, the optimal estimate for the interior for any $\eta > 0$ is

$$\arg\min_{f_U} f^T \boldsymbol{\Delta} f = (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} f_L,$$

which is harmonic. For any harmonic function f,

$$\begin{aligned} f^T \boldsymbol{\Delta} f &= f_L^T \boldsymbol{\Delta}_{LL} f_L + 2 f_L^T \boldsymbol{\Delta}_{LU} f_U + f_U^T \boldsymbol{\Delta}_{UU} f_U \\ &= f_L^T \boldsymbol{\Delta}_{LL} f_L - 2 f_L^T \boldsymbol{\Delta}_{LU} \boldsymbol{\Delta}_{UU}^{-1} \boldsymbol{\Delta}_{UL} f_L + f_L^T \boldsymbol{\Delta}_{LU} \boldsymbol{\Delta}_{UU}^{-1} \boldsymbol{\Delta}_{UL} f_L \\ &= f_L^T \boldsymbol{\Delta}_{LL}^{\star} f_L, \end{aligned}$$

so Optimization (18) produces a harmonic function with boundary solving

$$\min_{f_L} L(Y_L, f_L) + \eta f_L^T \boldsymbol{\Delta}_{LL}^{\star} f_L, \tag{19}$$

or equivalently

$$\min_{f_L} \underbrace{L(Y_L, f_L) + \eta f_L^T \widetilde{\boldsymbol{\Delta}}_{LL} f_L}_{\text{Supervised Objective}} + \underbrace{\eta f_L^T \boldsymbol{\Delta}_{LUL} f_L}_{\text{Offset}}.$$

Furthermore, under Optimization (19) with a finite loss $L(\cdot,\cdot)$, the clamped estimate of $f_L = Y_L$ is optimal for all $\eta > 0$ if and only if $Y_L \in \mathcal{N}(\boldsymbol{\Delta}_{LL}^{\star})$. In general, the clamped harmonic estimator is not necessarily optimal among harmonic estimators.

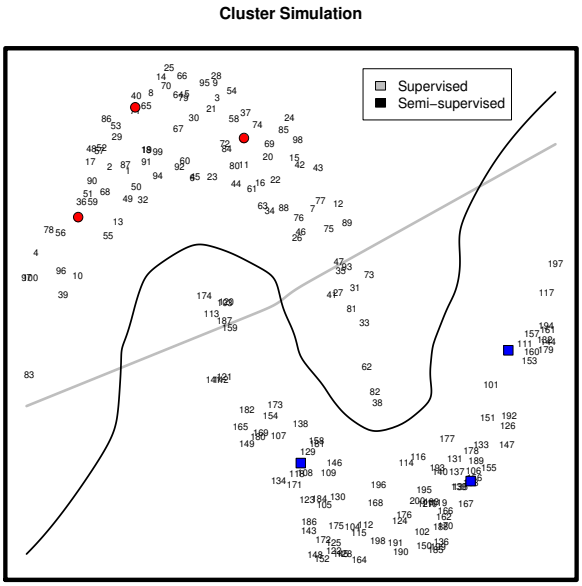


Figure 2: A "two moons" data set with $|L| = 6$ and $|U| = 200$. Label ● $= -1$ and ■ $= 1$.

4.2 Impact of Supervised Kernel Smoothing on Harmonic Estimators

Further examination of the cluster assumption is had by comparing the supervised kernel smoother (Section 3.1) to the semi-supervised harmonic estimator (Section 3.3). A goal is to understand why an observation $i \in U$ would sacrifice its own supervised estimate in favor of the cluster. Take the "two moons" example in Figure 2 that includes supervised and semi-supervised boundaries (see Remark 5). Focus on observation 38 in the downward pointing horn on right. According to the supervised rule this observation is ■ with probability 1. The semi-supervised prediction $f_{U_{38}} = -0.42$ is ● with probability 0.7, so the supervised estimate is overturned in favor of the cluster.

Any harmonic estimator with boundary f_L has the form $f_U = (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} f_L$. Assume $\widetilde{\boldsymbol{D}}_{UL} \succ 0$, so the supervised estimator exists (see Remark 4). Also, generalize the supervised predictions to $\widetilde{f}_U = \widetilde{\boldsymbol{S}}_{UL} f_L$, which we refer to as *soft supervised estimates*. Matrix $(\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL}$ is the product of the $|U| \times |U|$ stochastic matrix $(\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL}$ and the $|U| \times |L|$ supervised prediction matrix $\widetilde{\boldsymbol{S}}_{UL} = \widetilde{\boldsymbol{D}}_{UL}^{-1} \boldsymbol{W}_{UL}$, that is,

$$\begin{aligned} f_U &= (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} f_L && (20) \\ &= (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL} \widetilde{\boldsymbol{S}}_{UL} f_L \\ &= (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL} \widetilde{f}_U. && (21) \end{aligned}$$

Equation (21) shows that any semi-supervised harmonic function is a probability weighted average of the soft supervised estimators of U, that is,

$$f_{U_i} = \sum_{j \in U} \boldsymbol{P}_{ij} \widetilde{f}_j,$$

where the weights come from the stochastic matrix

$$\boldsymbol{P} = (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL}. \qquad (22)$$

Determining which soft supervised predictions $\widetilde{f}_U$ get the larger probability weights in the semi-supervised predictions f_U makes practical sense. Such a determination is possible if one relates the stochastic matrix $\boldsymbol{P}$ in Equation (22) to an absorbing Markov chain probability model (Doyle and Snell, 1984).

Consider the $|U|+1$ state Markov chain with transition matrix

$$\begin{pmatrix} \boldsymbol{S}_{UU} & (\boldsymbol{I}-\boldsymbol{S}_{UU})\,\vec{1} \\ \vec{0}^T & \vec{1} \end{pmatrix}.$$

Boundary L is treated as an *absorbing state*, and the harmonic estimator of the interior is

$$f_{U_i} = e_i^T f_U = \left(\sum_{k=0}^{\infty} e_i^T \boldsymbol{S}_{UU}^k \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL} \right) \widetilde{f}_U \text{ with elementary vector } e_i. \tag{23}$$

Each term in geometric series from Display (23) is the probability of a particular sequence of transitions with a given starting point in the absorbing Markov Chain probability model.

1: The first transition absorption to L starting from $i \in U$ is the $1 \times |U|$ row vector $e_i^T \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL}$, which has one non-zero entry. This non-zero, column i entry is the probability that a chain starting at unlabeled state i is absorbed into L at the first transition. This probability is large if unlabeled case $i \in U$ has more total similarity with cases in L than that with cases in U.

2: The second transition absorption to L from $j \in U$ starting from $i \in U$ is the row vector $e_i^T \boldsymbol{S}_{UU} \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL}$. Its jth column entry is the probability that a chain starting at unlabeled state i goes to unlabeled state j at first transition and is absorbed into L at the second transition.

...

k: The k'th transition absorption to L from $j \in U$ starting from $i \in U$ is the jth column entry of row vector $e_i^T \boldsymbol{S}_{UU}^{k-1} \boldsymbol{D}_{UU}^{-1} \widetilde{\boldsymbol{D}}_{UL}$. It is the probability that a chain starting at $i \in U$ goes $k-1$ transitions in U ending at some state $j \in U$ before being absorbed into L at the kth transition.

By Equation (23), the probability weight on soft supervised prediction $j \in U$ in semi-supervised prediction $i \in U$ is just the probability that a chain starting at $i \in U$ is absorbed from $j \in U$. Therefore, the soft supervised predictions for $j \in U$ that are "strongly adjacent" to observations in L carry the majority of the weight.

Back to Figure 2 for case 38. The top ten cases, that is, 72, 129, 84, 69, 74, 71, 36, 108, 20, and 59, carry 68% of the weight in the semi-supervised prediction of case 38, and each is close to a labeled observation. This "top ten" provides the approximation $\sum_{i=1}^{200} \boldsymbol{P}_{38,j} \widetilde{f}_{U_j} \approx \sum_{i=1}^{10} \boldsymbol{P}_{38(j)} \widetilde{f}_{U_{(j)}} = -0.38$ of the semi-supervised estimate, where (j) is the column of $\boldsymbol{P}$ containing its jth largest value in the 38th row. Hence the label prediction for observation 38 is already determined as -1 from this "top 10" because the combined weight of the other 190 cases at 32% is not enough to reverse the sign -0.38 given a ± 1 labeling. Furthermore, the supervised estimate for observation 38 is 68'th in the order with weight of only 0.002 or 0.2% in its very own semi-supervised prediction.

Remark 4 *"Assumption"* $\widetilde{\boldsymbol{D}}_{UL} \succ 0$ *is not necessary. If* $\widetilde{\boldsymbol{D}}_{UL} \not\succ 0$*, there exists* $i \in U$ *such that* $\widetilde{\boldsymbol{D}}_{UL_{ii}} = 0$*, and the supervised estimate does not exist for such i. This does not affect Equation*

(20), but is required in Factorization (21). Let $\widetilde{\boldsymbol{D}}_{UL}^{+}$ *be the diagonal generalized inverse of* $\widetilde{\boldsymbol{D}}_{UL}$ *with the same number of zero entries. If* $\widetilde{\boldsymbol{D}}_{UL}^{+}$ *is substituted in place of the nonexistent* $\widetilde{\boldsymbol{D}}_{UL}^{-1}$ *so that nonexistent soft supervised estimators are set to* $\widetilde{f}_{U_i} = \left(\widetilde{\boldsymbol{D}}_{UL}^{+}\boldsymbol{W}_{UL}f_L\right)_i = 0$*, Factorization (21) and its ensuing interpretation hold.*

5. Regularized Joint Harmonic Functions

Briefly consider the case when the response y is observed for all n observations. The Nadaraya-Watson kernel estimator $f = \boldsymbol{S}y$ results if functional $(y-f)^T\boldsymbol{W}(y-f)+f^T\boldsymbol{\Delta}f$ is minimized. In the semi-supervised setting when Y_U is missing, we replace y with the training response $Y(Y_U)$ from Display (3) and jointly optimize for both f and Y_U. In particular, the *regularized joint harmonic estimator* is the solution to

$$\min_{Y_U,f}\ (Y(Y_U)-f)^T\boldsymbol{W}(Y(Y_U)-f)+f^T\boldsymbol{\Delta}f+\gamma Y_U^TY_U \quad \text{(Joint Optimization Problem).} \tag{24}$$

The regularized joint harmonic estimator, given in Proposition 12, includes an estimator for both Y_U and f. The form of the f portion of this estimator is established as harmonic when $\gamma=0$ in Section 5.1. Discussion of the stabilizing effect due to the additional term $\gamma Y_U^TY_U$ in the context of the Joint Optimization Problem (24) when $\gamma>0$ is deferred until Section 5.2.

Proposition 12 *Let* $\boldsymbol{W}\succeq 0$*. Assume* $(\boldsymbol{\Delta S})_{UU}\succ 0$ *when one selects* $\gamma=0$*; this additional assumption is not required when one selects some* $\gamma>0$*. The unique solution to the Joint Harmonic Optimization Problem (24) is* $(Y_U,f)=\left(\hat{Y}_{U_\gamma},\boldsymbol{S}Y\left(\hat{Y}_{U_\gamma}\right)\right)$*, where*

$$\hat{Y}_{U_\gamma}=-\left((\boldsymbol{\Delta S})_{UU}+\gamma\boldsymbol{I}\right)^{-1}(\boldsymbol{\Delta S})_{UL}Y_L.$$

Matrix $\boldsymbol{\Delta S}$ has many of the properties of $\boldsymbol{\Delta}$ from Section 3, for example, $\boldsymbol{\Delta}\vec{1}=\vec{0}$ and $\boldsymbol{\Delta S}\vec{1}=\vec{0}$. Moreover, it is easy to verify that $\mathcal{N}(\boldsymbol{\Delta S})=\mathcal{N}(\boldsymbol{\Delta})$. Proposition 11 establishes a result for the positive semi-definiteness of $\boldsymbol{\Delta S}$, which is analogous to $\boldsymbol{\Delta}$ and Proposition 7.[1]

Proposition 11 *If* $\boldsymbol{W}\succeq 0$ *then* $\boldsymbol{\Delta S}\succeq 0$*.*

By Proposition 11, $\boldsymbol{W}\succeq 0$ is a sufficient condition for the uniqueness of the joint harmonic estimator when $\gamma>0$, but the added condition $(\boldsymbol{\Delta S})_{UU}\succ 0$ from Proposition 12 is needed if $\gamma=0$. Case $\gamma=0$ is discussed further in Section 5.1, and case $\gamma>0$ in Section 5.2.

Remark 5 *The prediction of a novel case given its nonnegative similarities* $(w_1,\cdots,w_n)$ *and response estimate* $\hat{Y}_{U_\gamma}$ *is computed from the Nadaraya-Watson kernel based function*

$$\breve{h}(w_1,\cdots,w_n)=\frac{\sum_{i\in L\cup U}w_iY_i\left(\hat{Y}_{U_\gamma}\right)}{\sum_{i\in L\cup U}w_i},$$

where $\breve{h}:\mathbb{R}^n\to\mathbb{R}$*. Finding the points in* $\mathbb{R}^n$ *that satisfy* $\breve{h}(w_1,\cdots,w_n)=0$ *is how one finds boundaries like those superimposed on Figure 2.*

1. Proposition 11 is used to prove Proposition 12 in Appendix A, but order was reversed here for presentation.

5.1 Joint Harmonic Estimator $\gamma = 0$

Here the joint harmonic function requires $(\boldsymbol{\Delta S})_{UU} \succ 0$ for its uniqueness (see Proposition 12). Results to come later in this section show that its boundary estimator is built on the unlabeled-unlabeled Schur complements of $\boldsymbol{W}$ and $\boldsymbol{\Delta}$ (refer to Section 2). First, Proposition 16 establishes an equivalence between these Schur complements and $(\boldsymbol{\Delta S})_{UU} \succ 0$.

Proposition 16 *If $\boldsymbol{W} \succeq 0$ then*

$$(\boldsymbol{\Delta S})_{UU} \succ 0 \iff \boldsymbol{W}_{UU} \succ 0,\ \boldsymbol{\Delta}_{UU} \succ 0, \text{ and } (\boldsymbol{W}^{\star}_{LL} + \boldsymbol{\Delta}^{\star}_{LL}) \succ 0.$$

Conditions from Proposition 16 are necessary and sufficient for the existence of the smoother

$$\boldsymbol{\Gamma}_{LL} = (\boldsymbol{W}^{\star}_{LL} + \boldsymbol{\Delta}^{\star}_{LL})^{-1} \boldsymbol{W}^{\star}_{LL} \quad \text{(Joint Harmonic Smoother)}, \tag{25}$$

and Theorem 18 states that smoother $\boldsymbol{\Gamma}_{LL}$ is that for the joint harmonic estimator.

Theorem 18 *Let $\boldsymbol{W} \succeq 0$, and assume that $\boldsymbol{\Gamma}_{LL}$ exists. The solution to the Joint Harmonic Optimization Problem (24) with $\gamma = 0$ has*

$$f = \begin{pmatrix} f_L \\ (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} f_L \end{pmatrix} = \begin{pmatrix} \boldsymbol{\Gamma}_{LL} \\ (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} \boldsymbol{\Gamma}_{LL} \end{pmatrix} Y_L,$$

so f is in-fact harmonic.

The work connecting the interior of a harmonic estimator to supervised estimators in Section 4.2 now applies to the joint harmonic estimator, that is, in particular recall $f_U = \boldsymbol{P}\widetilde{\boldsymbol{S}}_{UL}\boldsymbol{\Gamma}_{LL}Y_L$ with $\boldsymbol{P}$ from Display (22). One can view $\boldsymbol{\Gamma}_{LL}$ as a filter between the response Y_L and the supervised prediction smoother $\widetilde{\boldsymbol{S}}_{UL}$, which provides additional robustness for misspecified responses over that of using Y_L directly to form supervised predictions.

The boundary estimator is equivalently expressed as the solution to

$$\min_{f_L} (Y_L - f_L)^T \boldsymbol{W}^{\star}_{LL} (Y_L - f_L) + f_L^T \boldsymbol{\Delta}^{\star}_{LL} f_L. \tag{26}$$

Optimization (26) provides an interesting example of the labeled loss optimization problem from Display (18), where $\boldsymbol{W}^{\star}_{LL}$ allows unlabeled data to influence the weighted squared error loss functional independent of f_U. Hence, the harmonic result for labeled loss is still preserved, but the loss function is not independent of the unlabeled data. This also shows how this estimator generalizes the supervised case by replacing $\boldsymbol{W}_{LL}$ with $\boldsymbol{W}^{\star}_{LL}$ and $\widetilde{\boldsymbol{\Delta}}_{LL}$ with $\boldsymbol{\Delta}^{\star}_{LL}$. Furthermore, since $\boldsymbol{\Gamma}_{LL} = \boldsymbol{I} - (\boldsymbol{W}^{\star}_{LL} + \boldsymbol{\Delta}^{\star}_{LL})^{-1} \boldsymbol{\Delta}^{\star}_{LL}$,

$$\boldsymbol{\Delta}^{\star}_{LL} \nu = \vec{0} \iff \boldsymbol{\Gamma}_{LL} \nu = \nu,$$

so the joint harmonic estimator is a *cluster assumption classifier* (refer to Section 4.1). Proposition 19 provides further insight on the smoothing properties of $\boldsymbol{\Gamma}_{LL}$.

Proposition 19 *If $\boldsymbol{W} \succeq 0$ and $\boldsymbol{\Gamma}_{LL}$ exists then each eigenvalue of $\boldsymbol{\Gamma}_{LL}$ is an element of $[0,1]$.*

The above results for smoother $\boldsymbol{\Gamma}_{LL}$ are weaker than those for the stochastic semi-supervised smoother $\boldsymbol{M}_{LL}$ from Figure 1. In general, $\boldsymbol{\Gamma}_{LL}$ is not stochastic, although it was stochastic in nearly every numerical example we considered. In cases when $\boldsymbol{\Gamma}_{LL}$ is stochastic, the stronger condition that $|e_i^T f_U| \leq |e_i^T Y_L|$ holds, by the maximum principle of harmonic functions (Doyle and Snell, 1984).

In applications such as those in Sections 6 and 7, assumptions for the uniqueness of the $\gamma = 0$ joint harmonic estimator are not likely to be satisfied. These assumptions are especially sensitive to circumstances where $\boldsymbol{W}$ is generated from $\boldsymbol{X}$ with a kernel function set to small λ. The breakdown tends to worsen when $|L|/n$ is small. On the other hand, the $\gamma > 0$ regularized joint harmonic estimators in Section 5.2 elegantly relax these assumptions by modifying the Schur complements on the right of Display (25).

5.2 Regularized Joint Harmonic Estimators $\gamma > 0$

If the Joint Optimization Problem (24) is regularized with some $\gamma > 0$, the resulting joint estimator is unique. This estimator is built off of "regularized Schur complements"

$$\boldsymbol{W}^{\star}_{LL_\gamma} = \boldsymbol{W}_{LL} - \boldsymbol{W}_{LU}\boldsymbol{W}^{-}_{UU_\gamma}\boldsymbol{W}_{UL}, \tag{27}$$

$$\boldsymbol{\Delta}^{\star}_{LL_\gamma} = \boldsymbol{\Delta}_{LL} - \boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}^{-}_{UU_\gamma}\boldsymbol{\Delta}_{UL}, \tag{28}$$

where the "regularized inverses"

$$\boldsymbol{W}^{-}_{UU_\gamma} = \left(\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} + \gamma\boldsymbol{I}\right)^{-1}\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right)^{T}, \tag{29}$$

$$\boldsymbol{\Delta}^{-}_{UU_\gamma} = \left(\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{S}^{T}_{UU}. \tag{30}$$

If $\gamma = 0$, $\rho(\boldsymbol{S}_{UU}) < 1$, and $\rho\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right) < 1$, then $\boldsymbol{W}^{-}_{UU_0} = \boldsymbol{W}^{-1}_{UU}$ and $\boldsymbol{\Delta}^{-}_{UU_0} = \boldsymbol{\Delta}^{-1}_{UU}$, so regularized Schur complements in Displays (27) and (28) simplify to the Schur complements on the right of Display (25). It is also easily verified that

$$\boldsymbol{\Gamma}_{LL_\gamma} = \left(\boldsymbol{W}^{\star}_{LL_\gamma} + \boldsymbol{\Delta}^{\star}_{LL_\gamma}\right)^{-1}\boldsymbol{W}^{\star}_{LL_\gamma} \quad \text{(Regularized Joint Smoother)}$$

exists for any $\gamma > 0$. Theorem 21 extends Theorem 18 from $\gamma = 0$ to $\gamma > 0$.

Theorem 21 *Let $\boldsymbol{W} \succeq 0$. Let f_γ denote the solution to the Joint Harmonic Optimization Problem (24) with $\gamma > 0$. Then*

$$f_\gamma = \begin{pmatrix} \boldsymbol{\Gamma}_{LL_\gamma} \\ -\left(\boldsymbol{\Delta}^{-}_{UU_\gamma}\right)^{T}\boldsymbol{\Delta}_{UL}\boldsymbol{\Gamma}_{LL_\gamma} + \left(\boldsymbol{I} - \left(\boldsymbol{\Delta}^{-}_{UU_\gamma}\right)^{T}\boldsymbol{\Delta}_{UU}\right)\boldsymbol{S}_{UL} \end{pmatrix} Y_L.$$

The Theorem 21 decomposition is a compromise between the semi-supervised harmonic estimator (labeled connective) and supervised kernel estimator (labeled adjacent)

$$f_{U_\gamma} = \underbrace{-\left(\boldsymbol{\Delta}^{-}_{UU_\gamma}\right)^{T}\boldsymbol{\Delta}_{UL}f_{L_\gamma}}_{\text{Harmonic Part}} + \underbrace{\left(\boldsymbol{I} - \left(\boldsymbol{\Delta}^{-}_{UU_\gamma}\right)^{T}\boldsymbol{\Delta}_{UU}\right)\boldsymbol{S}_{UL}Y_L}_{\text{Supervised Part}}.$$

In the case of $\gamma = 0$, the harmonic part reduces to the harmonic estimator, and the supervised part equals zero. On the other extreme, as $\gamma \to \infty$, the harmonic part converges to zero, while the supervised part has limit

$$f_\gamma \to f_\infty = \boldsymbol{S}Y\left(\vec{0}\right) = \begin{pmatrix} \boldsymbol{S}_{LL} \\ \boldsymbol{S}_{UL} \end{pmatrix} Y_L = \begin{pmatrix} \boldsymbol{Q}_L\widetilde{\boldsymbol{S}}_{LL} \\ \left(\boldsymbol{I} - \boldsymbol{Q}_U\right)\widetilde{\boldsymbol{S}}_{UL} \end{pmatrix} Y_L = \begin{pmatrix} \boldsymbol{Q}_L\widetilde{f}_L \\ \left(\boldsymbol{I} - \boldsymbol{Q}_U\right)\widetilde{f}_U \end{pmatrix}, \tag{31}$$

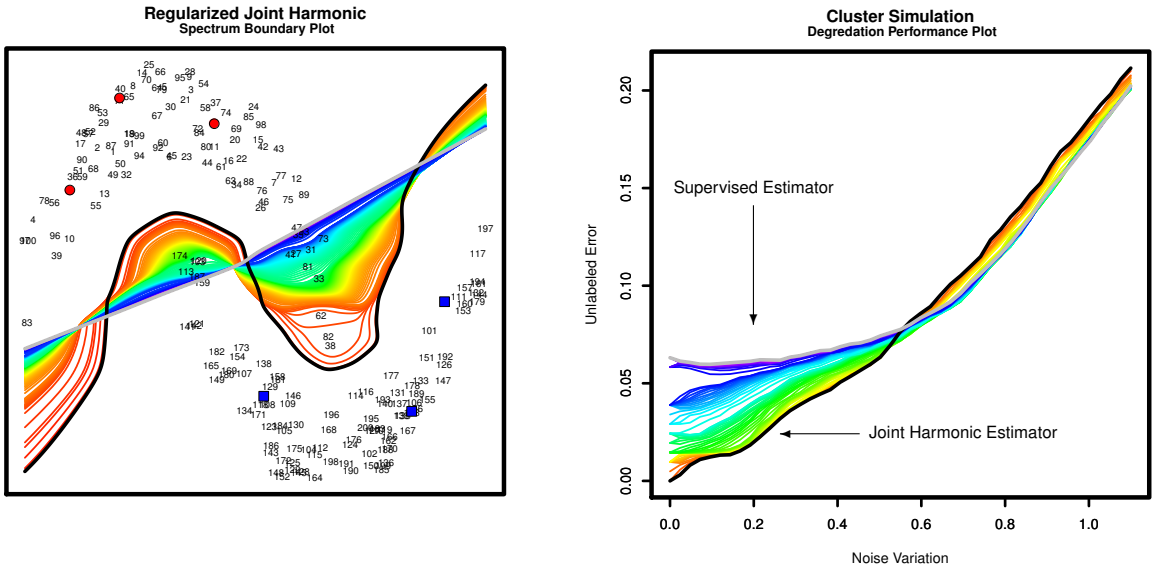


Figure 3: The "two moons" data from Figure 2 with regularized joint harmonic classification boundary curves on left. Noise degradation study on right. Black: *gamma* $= 0$ (harmonic extreme). Gray: $\gamma = \infty$ (supervised extreme). Rainbow spectrum: ordered by $\gamma \in (0, \infty)$.

where diagonal matrix $\boldsymbol{Q}_U$ has $\boldsymbol{Q}_{U_{ii}} = \sum_{j \in U} \boldsymbol{W}_{ij} / \sum_{j \in L \cup U} \boldsymbol{W}_{ij}$ for $i \in U$ and $\boldsymbol{Q}_L$ is defined analogously on L (apply Remark 4 when entries in $\widetilde{f}_U$ do not exist). Each estimator is a multiple of the supervised case by the right of Equation (31), so $\lim_{\gamma \to \infty} \text{sign}(f_{\gamma_i}) = \text{sign}(\widetilde{f}_i)$ for every i in the context of a classification problem with $Y_L \in \{-1, 1\}^{|L|}$.

The "two moons" data from Figure 2 are now revisited in Figure 3. The black joint harmonic function ($\gamma = 0$) and the gray supervised extreme ($\gamma = \infty$) borders in the left panel of Figure 3 correspond to the harmonic and supervised borders in Figure 2 as expected. The rainbow spectrum of borders rely less on the interior network and more on local supervised estimates as γ increases. Now, suppose the "two moons" data were instead observed with noise around each observation. Independent random samples from $\text{N}(0, \sigma^2)$ were added to each coordinate after scaling each axis in the left panel to sample standard deviation one. The regularized joint harmonic estimate was computed for each γ and σ over a grid, and unlabeled errors were recorded over this grid assuming the "truth" of a constant labeling by moon in the $\sigma = 0$ noiseless data on left. This was repeated 50 times, and average unlabeled error rates versus noise variation σ are plotted by γ in the right panel of Figure 3. While the joint harmonic function and the supervised solution are optimal for small and large σ, compromise solutions are best for data with an intermediate level of noise. Overall, the regularized joint harmonic estimator is a compromise between the harmonic estimator (which emphasizes unlabeled connectivity to labeled cases) and the supervised estimator (which requires unlabeled adjacency to labeled cases).

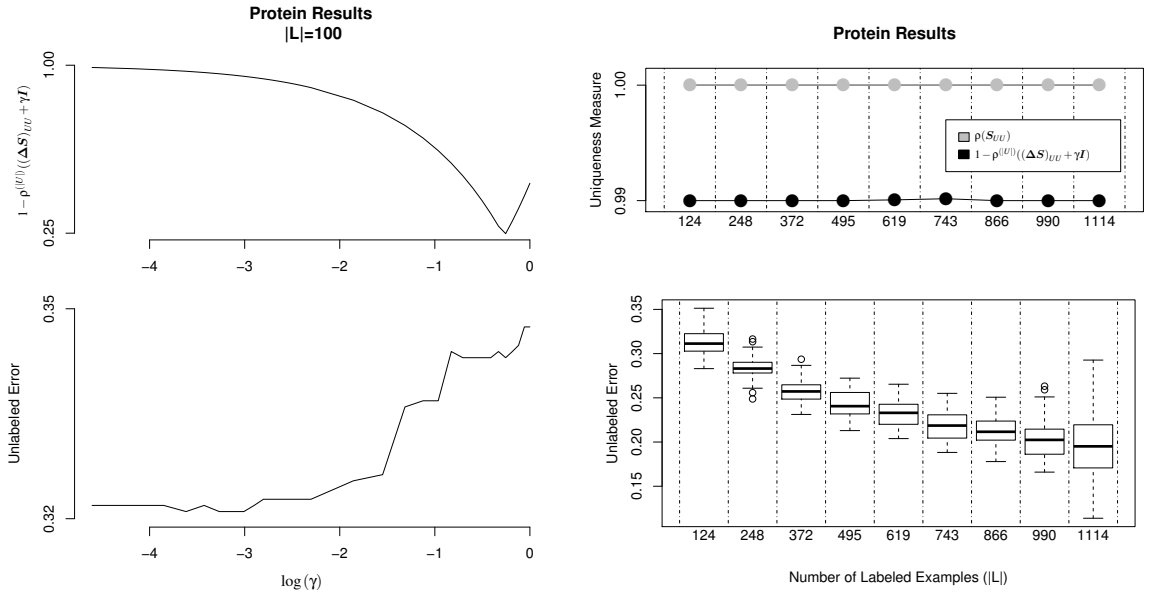


Figure 4: Regularized joint harmonic analyses of the protein data. Left: Uniqueness condition (top) and performance measure (bottom) versus regularization parameter $\log(\gamma)$ with a labeled set of size $|L| = 100$. Right: Uniqueness measure (top) and performance (bottom) for each of 50 replicates at each $|L|$ tested.

5.3 Joint Training Connections

The regularized joint harmonic estimator is the solution to a particular version of a *generalized joint training* optimization problem

$$\min_{Y_U, f} L(Y(Y_U), f) + \eta J_1(f) + \gamma J_2(Y_U) \tag{32}$$

with $L(y, f)$ a loss function, $J_1(f) \geq 0$ a penalty term independent of Y_U with $\eta \geq 0$, and $J_2(Y_U) \geq 0$ a penalty term independent of f with $\gamma \geq 0$. It is clear how to choose $L(\cdot,\cdot)$, $J_1(\cdot)$, and $J_2(\cdot)$ so that the generalized problem from Display (32) simplifies to the problem in Display (24). The S^3VM (Chapelle et al., 2006a) is approximated by setting $L(\cdot,\cdot)$ as a diagonally weighted hinge loss function with $L(Y(Y_U), f) = c_1 \sum_{i \in L}(1 + Y_i f_i)_+ + c_2 \sum_{i \in U}(1 + Y_i f_i)_+$ for $c_1, c_2 \in \mathbb{R}^+$, optimizing Y_U in a binary space, setting $J_1(f)$ as a quadratic ambient penalty, and forcing $\gamma = 0$. In this case, $\sum_{i \in U}(1 + Y_i f_i)_+$ is referred to as an interplay penalty between Y_U and f_U. The SSVM and SPSI algorithms are also construed as approximations of Optimization (32) (Wang and Shen, 2007). Lastly, linear joint training was proposed in Culp (2013) to extend the elastic net and other linear approaches into the semi-supervised setting.

6. Protein Interaction Data

Data on $n = 1237$ proteins from yeast organisms were collected. Each of 13 systems was used to detect the presence of protein-to-protein interactions (Kui et al., 2002). Adjacencies in $\boldsymbol{W}$ are taken

to be the proportion of systems detecting an interaction, so

$$\boldsymbol{W}_{ij} = \begin{cases} \frac{1}{13}\sum_s I_{\{\text{system } s \text{ detected an interaction between proteins } i,j\}} & i \neq j \\ 1 & i = j. \end{cases}$$

An important yet difficult problem is to classify whether or not a protein is located on the nucleus of a cell (Yamanishi et al., 2004). A number of analyses of $\boldsymbol{W}$ using the regularized joint harmonic estimator are presented in Figure 4. All 1237 proteins were included in each analysis, but the definition of the boundary was altered. The clamped harmonic and joint harmonic approaches are singular in each of these analyses, whereas the regularization strategy posed for the joint harmonic estimator provides the practical benefit of a well-defined classifier with a unique solution. Furthermore, the protein interaction graph $\boldsymbol{W}$ was observed directly, so there is no tuning parameter for either harmonic estimator.

Boundary L is 100 randomly selected proteins in the left panels of Figure 4. Since $\rho(\boldsymbol{S}_{UU}) = 1$, any harmonic estimator is singular. On the other hand, the regularized joint harmonic estimator is applicable with large enough γ so that $(\boldsymbol{\Delta S})_{UU} + \gamma \boldsymbol{I}$ is invertible, that is, when $\rho^{(|U|)}((\boldsymbol{\Delta S})_{UU} + \gamma \boldsymbol{I}) > 0$ in the top panel. The corresponding unlabeled error performance as a function of $\log(\gamma)$ is plotted in the bottom panel.

Consider now the analyses in the right panels of Figure 4. Proportion $|L|/n$ was varied from 0.1 to 0.9 by 0.1, and an analysis like that on the left was run for each of 50 randomly selected boundary sets at each $|L|$. The top right panel shows that the spectral radius uniqueness assumption was violated for any harmonic estimator, for example, the clamped or $\gamma = 0$, whereas regularization of the joint harmonic approach identified a well-defined classifier. The corresponding testing errors indicate a trend toward improved performance as the size of the labeled set increases in the bottom panel.

7. Machine Learning Data Sets

A comparison of procedures was based on three data sets from the UCI repository (Frank and Asuncion, 2010), that is, the ionosphere data set with $n = 351$ observations, thyroid data $n = 215$, and breast cancer data $n = 699$, and a publicly available pharmaceutical solubility data set with $n = 5631$ (Izenman, 2008). Missing values within the solubility data were handled by mean imputation. The $|L \cup U| \times |L \cup U|$ matrix $\boldsymbol{W}$ was computed from $\boldsymbol{X}$ feature data using the Gaussian kernel function, that is, $\boldsymbol{W}_{ij} = K_\lambda(x_i, x_j)$. Five-fold cross-validation was used to estimate $(\hat{\lambda}, \hat{\gamma})$ for the regularized joint harmonic function and $\hat{\lambda}$ for the clamped harmonic estimator. A semi-supervised SVM (S^3VM) with a linear kernel was also fit; its cost and gamma parameters were estimated using cross-validation with the *svm.tune* function from **R** library *e1071* (R Core Team, 2012; Meyer et al., 2012).

A transductive comparison is provided by Figure 5. The ionosphere and thyroid data were each randomly partitioned into L and U sets 50 times for each $|L| = 10, 20, 30, 40, 50$, and the techniques were all run on the same L and U partitions. The top and middle panels of Figure 5 summarize a particular example with $|L| = 20$ from the corresponding bottom panel. The clamped harmonic estimator is computationally singular and cannot be computed when $\rho(\boldsymbol{S}_{UU}) \approx 1$ (see Remark 6). This occurs for any $\lambda < 0.3$, that is, $\log(\lambda) < -1.2$, in the ionosphere application and for any $\lambda < 0.2$, that is, $\log(\lambda) < -1.6$, in the thyroid application. The joint harmonic estimator ($\gamma = 0$) requires the more stringent assumption $(\boldsymbol{\Delta S})_{UU} \succ 0$, and it was singular for all λ in the ionosphere

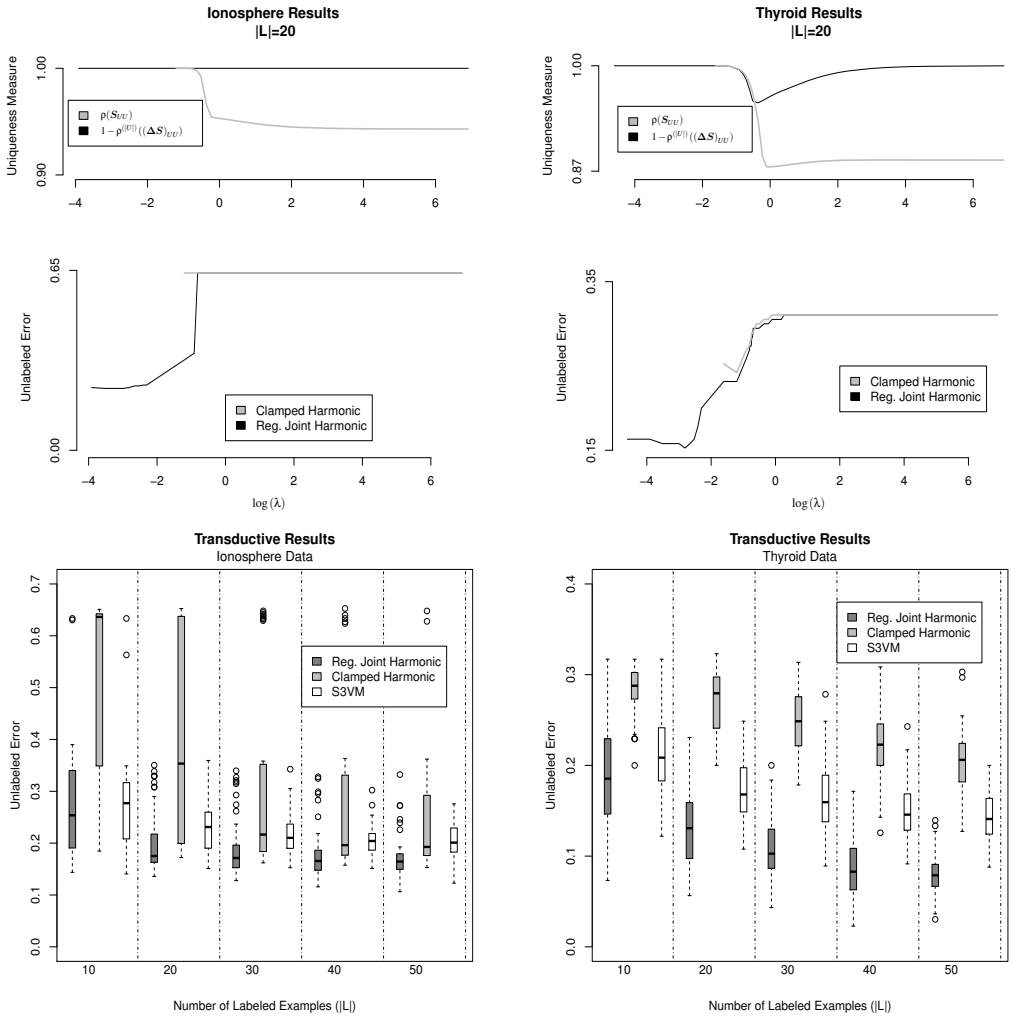


Figure 5: Transductive results for the ionosphere (left) and thyroid (right) data sets. Uniqueness measure (top) and unlabeled error performance (middle) each versus kernel parameter $\log(\lambda)$ for a particular analysis with $|L| = 20$ from the bottom panels. Unlabeled error rate performance (bottom) of the regularized joint harmonic, clamped harmonic, and S^3VM estimators for 50 randomly selected labeled sets L of each size $|L| = 10, 20, 30, 40, 50$.

application. However, estimates $\hat{\gamma} = 0.5$ and $\hat{\gamma} = 0.04$ in the ionosphere and thyroid applications were obtainable with the regularized joint harmonic estimator. Its access to a wider range of values λ, especially small λ, may yield substantial improvement in performance in other applications, like that seen in the bottom panels of Figure 5. As expected, a substantial performance gap exists between the regularized joint harmonic estimator and the clamped harmonic estimator. The S^3VM also outperformed the clamped harmonic estimator.

A semi-supervised comparison is provided by Figure 6. The data were first randomly partitioned into "seen" (25%) and "unseen" (75%) cases. The seen cases $L \cup U$ were then randomly partitioned into sets L and U of each size $|L| = 10, 20, 30, 40, 50$. The techniques were all run on the same

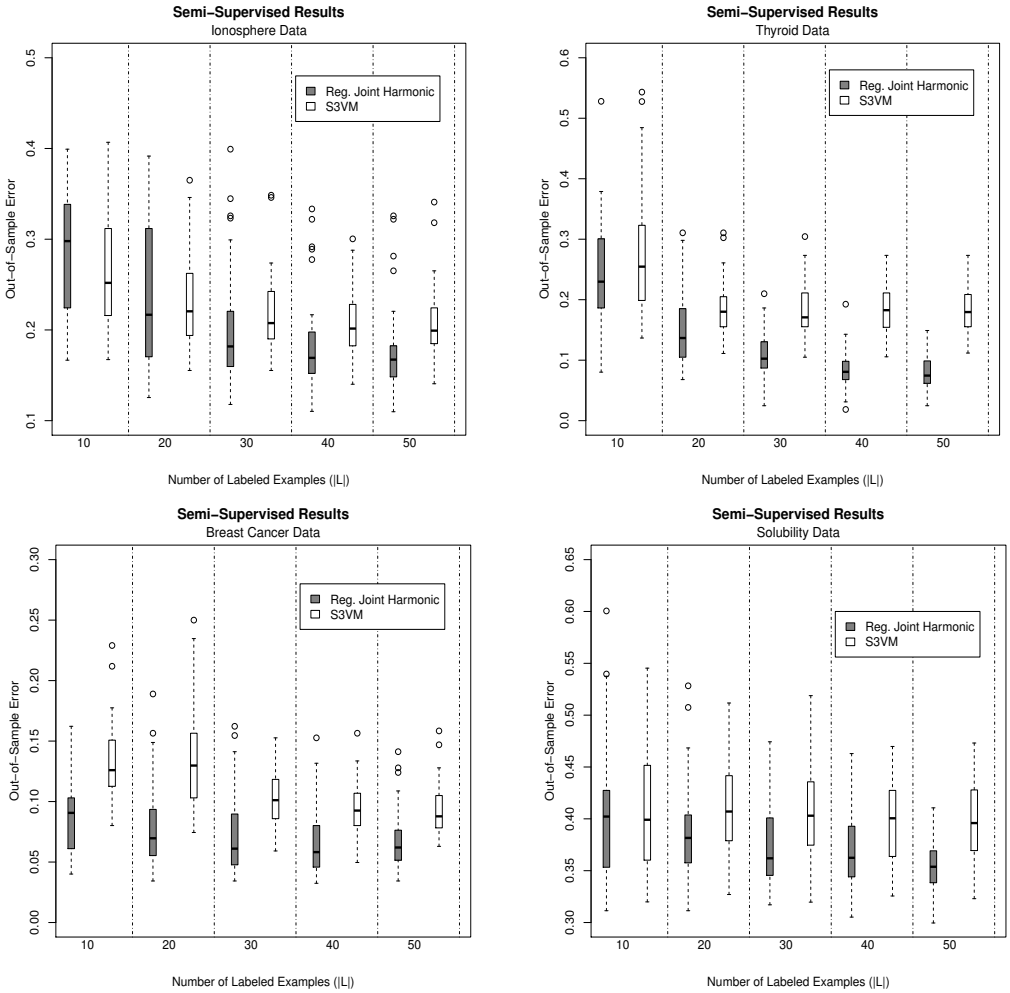


Figure 6: Semi-supervised out-of-sample error rate performance of the regularized joint harmonic and S^3VM estimators on four publicly available data sets. Each randomly obtained out-of-sample extension was 75% of cases. The other 25% were treated as $L \cup U$. Labeled sets L of each size $|L| = 10, 20, 30, 40, 50$ were also obtained randomly prior to cross-validation. This entire process was repeated 50 times.

"unseen," L, and U partitions, and the entire process was repeated 50 times. The clamped harmonic estimator is no longer applicable. Semi-supervised performance comparisons (Figure 6) of the regularized joint harmonic approach to the S^3VM are consistent with the transductive case (Figure 5), and the variability of the error measure increased in the out-of-sample extension as expected. In short, Figures 5 and 6 include real, low labeled sample size, transductive and semi-supervised applications, and the competitive stature of our proposed regularized joint harmonic estimator holds.

Remark 6 *Assumptions for the uniqueness of the clamped harmonic and the regularized joint harmonic approaches depend on the denseness or sparseness of the $\boldsymbol{W}_{UL}$ component of similarity graph $\boldsymbol{W}$. Sparseness makes the needed eigenvalue conditions more difficult to satisfy. One might expect a*

more sparse $\boldsymbol{W}_{UL}$ component when the labeled set size $|L|$ is small relative to the unlabeled set size $|U|$. As kernel parameter λ decreases, the off-diagonal elements of $\boldsymbol{W}$ approach 0, and this forces computational zeros in matrix $\boldsymbol{W}_{UL}$ leading to less stable estimators for any harmonic estimator. This follows since $\boldsymbol{S}\vec{1} = \vec{1}$, and so if $\boldsymbol{S}_{UL}\vec{1} \approx \vec{0}$, then $\boldsymbol{S}_{UU}\vec{1} \approx \vec{1}$. Hence, $\rho(\boldsymbol{S}_{UU}) \approx 1$ in the sparse case. On the other hand, larger values of λ allow the potential for a denser $\boldsymbol{W}_{UL}$ component which potentially makes the eigenvalue assumptions less stringent. The parameter λ is estimated using five-fold cross-validation, which does not account for assumptions on $\boldsymbol{W}_{UL}$. Regularization within the joint harmonic approach has the key advantage of a unique estimator for any $\lambda > 0$.

8. Conclusion

Semi-supervised harmonic estimation for graph-based semi-supervised learning was examined theoretically and empirically. A cluster assumption classifier was also defined, and it was shown that such classifiers assign labels to data that conform to the cluster assumption in the logical manner. Harmonic functions with a well-chosen boundary are examples of cluster assumption classifiers. In addition, harmonic functions were shown to be weighted averages of local supervised estimators applied to the interior. This work further established that harmonic estimators rely primarily on connectivity within the unlabeled network to form predictions using local supervised estimators; supervised estimates near labeled cases are up-weighted while supervised estimates deep within the network are down-weighted. Another key contribution, the development of the regularized joint harmonic function approach, used a joint optimization criterion with regularization to automate the trade-off between labeled connectivity versus labeled adjacency. Empirical results demonstrated the practical benefit gained by regularization of joint harmonic estimation.

Acknowledgments

The authors thank the AE and anonymous referees for their useful comments and suggestions. The work of Mark Vere Culp was supported in part by the NSF CAREER/DMS-1255045 grant. The work of Kenneth Joseph Ryan was supported in part by the U.S. Department of Justice 2010-DD-BX-0161 grant. The opinions and views expressed in this paper are those of the authors and do not reflect the opinions or views at either the NSF or the U.S. Department of Justice.

Appendix A. Proofs

Proofs of Lemmas, Propositions, and Theorems follow.

A.1 Problem Set-Up

Proposition 7 *Laplacian $\boldsymbol{\Delta} \succeq 0$.*

Proof Matrix $\boldsymbol{\Delta}$ satisfies $\boldsymbol{\Delta}_{ii} = \sum_{k=1}^{n} \boldsymbol{W}_{ik} I_{\{i \neq k\}} \geq \boldsymbol{W}_{ij} = -\boldsymbol{\Delta}_{ij} \geq 0$ for each $i \neq j$, and such symmetric, diagonally dominant Z-matrices are positive semi-definite. ■

Proposition 8 *If $\boldsymbol{W} \succeq 0$ then each eigenvalue of $\boldsymbol{S} = \boldsymbol{D}^{-1}\boldsymbol{W}$ is an element of $[0,1]$.*

Proof Matrices $\boldsymbol{S}$ and $\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2} \succeq 0$ have the same eigenvalues, so the eigenvalues of $\boldsymbol{S}$ are bounded below by 0. Proposition 7 implies $\boldsymbol{D}^{-1/2}\boldsymbol{\Delta}\boldsymbol{D}^{-1/2} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2} \succeq 0$, so the eigenvalues of $\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}$ and hence $\boldsymbol{S}$ are also bounded above by 1. ■

Lemma 9 *If $\boldsymbol{W} \succeq 0$ then each eigenvalue of $\boldsymbol{S}_{UU}$ is an element of $[0,1]$.*

Proof Define $\boldsymbol{I}_U = \operatorname{diag}\left(\mathbf{1}_{\{i \in U\}}\right)$ based on the binary vector $\mathbf{1}_{\{i \in U\}} \in \mathbb{R}^{|L \cup U|}$. Matrices $\boldsymbol{S}_{UU}$ and $\left(\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}\right)_{UU} = \boldsymbol{I}_U\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}\boldsymbol{I}_U \succeq 0$ have the same eigenvalues, so

$$\rho(\boldsymbol{S}_{UU}) = \rho\left(\boldsymbol{I}_U\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}\boldsymbol{I}_U\right) \leq \rho\left(\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}\right) \leq 1,$$

where the second inequality was justified during the proof of Proposition 8. ■

Proposition 10 *If $\boldsymbol{W} \succeq 0$ then the following conditions are equivalent.*

(a) $\boldsymbol{\Delta}_{UU} \succ 0$.

(b) $\rho(\boldsymbol{S}_{UU}) < 1$.

(c) $\nu^T \widetilde{\boldsymbol{D}}_{UL}\nu > 0$ *for any non-zero* $\nu \in \mathcal{N}(\widetilde{\boldsymbol{D}}_{UU} - \boldsymbol{W}_{UU})$.

Proof [(a) $\Longleftrightarrow$ (b)]: This equivalence follows by taking inverses of $\boldsymbol{\Delta}_{UU} = \boldsymbol{D}_{UU}\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right)$. Condition (a) implies $\mathcal{N}(\boldsymbol{\Delta}_{UU}) = \left\{\vec{1}\right\}$, so condition (b) follows because the Lemma 9 upper bound of 1 for the largest eigenvalue of $\boldsymbol{S}_{UU}$ cannot be achieved. Condition (b) implies the existence of $(\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1}$ by a geometric matrix series, and so condition (a) follows.

[(a) $\Longleftrightarrow$ (c)]: Proposition 7 implies $\boldsymbol{\Delta}_{UU} \succeq 0$, so if $\nu \in \mathcal{N}\left(\widetilde{\boldsymbol{D}}_{UU} - \boldsymbol{W}_{UU}\right)$,

$$\nu^T\boldsymbol{\Delta}_{UU}\nu = \nu^T\widetilde{\boldsymbol{D}}_{UL}\nu + \nu^T\left(\widetilde{\boldsymbol{D}}_{UU} - \boldsymbol{W}_{UU}\right)\nu > 0 \iff \nu^T\widetilde{\boldsymbol{D}}_{UL}\nu > 0.$$

■

A.2 Regularized Joint Harmonic Functions

Proposition 11 *If $\boldsymbol{W} \succeq 0$ then $\boldsymbol{\Delta}\boldsymbol{S} \succeq 0$.*

Proof Define the matrix

$$\boldsymbol{V} = \begin{pmatrix} \boldsymbol{W} & \boldsymbol{W} \\ \boldsymbol{W} & \boldsymbol{D} \end{pmatrix}, \text{ and let } \nu = \begin{pmatrix} \nu_1 \\ \nu_2 \end{pmatrix} \in \mathbb{R}^{2|L \cup U|} \text{ with } \nu \neq \vec{0}. \tag{33}$$

Since $\nu^T\boldsymbol{V}\nu = \nu_1^T\boldsymbol{W}\nu_1 + \nu_1^T\boldsymbol{W}\nu_2 + \nu_2^T\boldsymbol{W}\nu_1 + \nu_2^T\boldsymbol{D}\nu_2 = (\nu_1+\nu_2)^T\boldsymbol{W}(\nu_1+\nu_2) + \nu_2^T\boldsymbol{\Delta}\nu_2 \geq 0$, the $\boldsymbol{D}$ block Schur complement of $\boldsymbol{V}$ is positive semi-definite, that is, $\boldsymbol{W} - \boldsymbol{W}\boldsymbol{D}^{-1}\boldsymbol{W} = \boldsymbol{\Delta}\boldsymbol{S} \succeq 0$. ■

Proposition 12 *Let $\boldsymbol{W} \succeq 0$. Assume $(\boldsymbol{\Delta S})_{UU} \succ 0$ when one selects $\gamma = 0$; this additional assumption is not required when one selects some $\gamma > 0$. The unique solution to the Joint Harmonic Optimization Problem (24) is $(Y_U, f) = \left(\hat{Y}_{U_\gamma}, \boldsymbol{SY}\left(\hat{Y}_{U_\gamma}\right)\right)$, where*

$$\hat{Y}_{U_\gamma} = -\left((\boldsymbol{\Delta S})_{UU} + \gamma \boldsymbol{I}\right)^{-1} (\boldsymbol{\Delta S})_{UL} Y_L.$$

Proof The solution is unique if the scores of the quadratic in (Y_U, f) objective function are non-degenerate. After some rearrangement, the scores with respect to Y_U and f are

$$\begin{aligned} \boldsymbol{S}_{UU}(\hat{Y}_{U_\gamma} - f_U) + \boldsymbol{S}_{UL}(Y_L - f_L) + \gamma \boldsymbol{D}_{UU}^{-1} \hat{Y}_{U_\gamma} &= \vec{0} \quad (34) \\ f(Y_U) &= \boldsymbol{SY}(Y_U), \quad (35) \end{aligned}$$

and plugging the f_U portion of Vector (35) into Unlabeled Score (34) produces

$$\begin{aligned} \boldsymbol{D}_{UU}^{-1}\left(\gamma \boldsymbol{I} + \boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} + \boldsymbol{\Delta}_{UL}\boldsymbol{S}_{LU}\right)\hat{Y}_{U_\gamma} &= -\boldsymbol{D}_{UU}^{-1}\left(\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UL} + \boldsymbol{\Delta}_{UL}\boldsymbol{S}_{LL}\right) Y_L \\ \hat{Y}_{U_\gamma} &= -\left((\boldsymbol{\Delta S})_{UU} + \gamma \boldsymbol{I}\right)^{-1} (\boldsymbol{\Delta S})_{UL} Y_L. \end{aligned}$$

Matrix $(\boldsymbol{\Delta S})_{UU} + \gamma \boldsymbol{I} \succ 0$ by Proposition 11 when $\gamma > 0$ and by assumption when $\gamma = 0$, so its inverse exists. Substitution of $Y_U = \hat{Y}_{U_\gamma}$ into Equation (35) results in $f = \boldsymbol{SY}\left(\hat{Y}_{U_\gamma}\right)$. ■

A.3 Joint Harmonic Estimator $\gamma = 0$

Lemma 13 *If $\boldsymbol{W} \succeq 0$ then $\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} = \boldsymbol{D}_{UU}\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right)\boldsymbol{S}_{UU} \succeq 0$. In addition,*

$$\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} \succ 0 \iff \rho(\boldsymbol{S}_{UU}) < 1 \text{ and } \rho\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right) < 1.$$

Proof In Display (33), substitute $\boldsymbol{W}_{UU}$ for $\boldsymbol{W}$ and $\boldsymbol{D}_{UU}$ for $\boldsymbol{D}$ and take $\nu \in \mathbb{R}^{2|U|}$. Then

$$\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} \succeq 0 \iff (\nu_1 + \nu_2)^T \boldsymbol{W}_{UU}(\nu_1 + \nu_2) + \nu_2^T \boldsymbol{\Delta}_{UU}\nu_2 \geq 0. \quad (36)$$

One can set $\nu_2 = \vec{0}$ or $\nu_1 + \nu_2 = \vec{0}$ such that $\nu \neq \vec{0}$, so both inequalities in Display (36) are strict if and only if $\boldsymbol{\Delta}_{UU} \succ 0$ and $\boldsymbol{W}_{UU} \succ 0$. Furthermore, $\boldsymbol{\Delta}_{UU} \succ 0 \iff \rho(\boldsymbol{S}_{UU}) < 1$ by Proposition 10, and $\boldsymbol{W}_{UU} \succ 0 \iff \rho\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right) < 1$ by Lemma 9. ■

Lemma 14 *Let $\boldsymbol{W} \succeq 0$. Also, assume $\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} \succ 0$, so $\boldsymbol{A} = \boldsymbol{S}_{LU}\boldsymbol{S}_{UU}^{-1}\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL}$ exists by Lemma 13. Then each eigenvalue of $\boldsymbol{A}$ is an element of $[0,1]$.*

Proof Each eigenvalue of $\boldsymbol{S}_{UU}$ is an element of $(0,1)$ by Lemma 9, since $\boldsymbol{W}_{UU} \succ 0$ rules out eigenvalues of 0 and $\boldsymbol{\Delta}_{UU} \succ 0$ eigenvalues of 1 by Proposition 10. Furthermore, the UU block Schur complements $\boldsymbol{\Delta}_{LL}^\star$ and $\boldsymbol{W}_{LL}^\star$ are each positive semi-definite, so

$$\boldsymbol{B}_1 = \boldsymbol{D}_{LL}^{-1/2}\left(\boldsymbol{W}_{LL}^\star + \boldsymbol{\Delta}_{LL}^\star\right)\boldsymbol{D}_{LL}^{-1/2} \succeq 0. \quad (37)$$

By assumption (and application of Lemma 13), $\boldsymbol{D}_{UU}\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UU}^{-1} \succ 0$, so since a row of $\boldsymbol{W}_{UL}$ could be all zeros,

$$\boldsymbol{B}_2 = \boldsymbol{D}_{LL}^{-1/2}\boldsymbol{W}_{LU}\left(\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{W}_{UL}\boldsymbol{D}_{LL}^{-1/2} \succeq 0.$$

Although tedious to establish, there is a simple relationship between $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$; that is,

$$\begin{aligned}
\boldsymbol{B}_2 &= \boldsymbol{D}_{LL}^{-1/2}\boldsymbol{W}_{LU}\boldsymbol{S}_{UU}^{-1}\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL}\boldsymbol{D}_{LL}^{-1/2} \\
&= \boldsymbol{D}_{LL}^{-1/2}\boldsymbol{W}_{LU}\boldsymbol{S}_{UU}^{-1}\boldsymbol{S}_{UL}\boldsymbol{D}_{LL}^{-1/2}+\boldsymbol{D}_{LL}^{-1/2}\boldsymbol{W}_{LU}\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL}\boldsymbol{D}_{LL}^{-1/2} \qquad (38) \\
&= \boldsymbol{D}_{LL}^{-1/2}\boldsymbol{W}_{LU}\boldsymbol{W}_{UU}^{-1}\boldsymbol{W}_{UL}\boldsymbol{D}_{LL}^{-1/2}+\boldsymbol{D}_{LL}^{-1/2}\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL}\boldsymbol{D}_{LL}^{-1/2} \\
&= \boldsymbol{I}-\boldsymbol{D}_{LL}^{-1/2}\left(\left(\boldsymbol{D}_{LL}-\boldsymbol{W}_{LL}-\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL}\right)+\left(\boldsymbol{W}_{LL}-\boldsymbol{W}_{LU}\boldsymbol{W}_{UU}^{-1}\boldsymbol{W}_{UL}\right)\right)\boldsymbol{D}_{LL}^{-1/2} \\
&= \boldsymbol{I}-\boldsymbol{B}_1,
\end{aligned}$$

where equality holds in Display (38) because $\boldsymbol{S}_{UU}^{-1}\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}=\boldsymbol{S}_{UU}^{-1}+\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}$.

The eigenvalues of $\boldsymbol{B}_2$ are bounded below by 0 because $\boldsymbol{B}_2 \succeq 0$ and bounded above by 1 because $\boldsymbol{B}_1 \succeq 0$ and $\boldsymbol{B}_2=\boldsymbol{I}-\boldsymbol{B}_1 \succeq 0$. This proof concludes by noting that $\boldsymbol{B}_2$ and $\boldsymbol{A}$ have the same eigenvalues since $\boldsymbol{B}_2\phi=\lambda\phi \iff \boldsymbol{A}\breve{\phi}=\lambda\breve{\phi}$, where $\breve{\phi}=\boldsymbol{D}_{LL}^{-1/2}\phi$. ■

Lemma 15 *If $\boldsymbol{W} \succeq 0$ then the following conditions are equivalent.*

(a) $(\boldsymbol{\Delta S})_{UU} \succ 0$.

(b) $\rho(\boldsymbol{S}_{UU})<1$, $\rho\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)<1$, *and* $\rho(\boldsymbol{A})<1$, *where* $\boldsymbol{A}=\boldsymbol{S}_{LU}\boldsymbol{S}_{UU}^{-1}\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL}$.

(c) $\boldsymbol{W}_{UU} \succ 0$, $\boldsymbol{\Delta}_{UU} \succ 0$, *and* $(\boldsymbol{W}_{LL}^{\star}+\boldsymbol{\Delta}_{LL}^{\star}) \succ 0$.

(d) $\boldsymbol{\Gamma}_{LL}=(\boldsymbol{W}_{LL}^{\star}+\boldsymbol{\Delta}_{LL}^{\star})^{-1}\boldsymbol{W}_{LL}^{\star}$ *exists.*

Proof [(a) $\iff$ (b)]: Matrix $(\boldsymbol{\Delta S})_{UU} \succeq 0$ by Proposition 11. Also,

$$(\boldsymbol{\Delta S})_{UU}=\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU}-\boldsymbol{W}_{UL}\boldsymbol{D}_{LL}^{-1}\boldsymbol{W}_{LU}$$

is the $\boldsymbol{D}_{LL}$ block Schur complement of

$$\boldsymbol{V}_2=\begin{pmatrix}\boldsymbol{D}_{LL} & \boldsymbol{W}_{LU} \\ \boldsymbol{W}_{UL} & \boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU}\end{pmatrix},$$

so condition (a) $\iff \boldsymbol{V}_2 \succ 0$. Hence, it suffices to show $\boldsymbol{V}_2 \succ 0 \iff$ condition (b). This follows because $\boldsymbol{V}_2 \succ 0 \iff$ the $\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU}$ block Schur complement of $\boldsymbol{V}_2$ is positive definite, that is, $\left(\boldsymbol{D}_{LL}-\boldsymbol{W}_{LU}\left(\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{W}_{UL}\right)=\boldsymbol{D}_{LL}(\boldsymbol{I}-\boldsymbol{A}) \succ 0$. Recall $(\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU})^{-1} \iff \rho(\boldsymbol{S}_{UU})<1$ and $\rho\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)<1$ by Lemma 13. Furthermore, the existence of $(\boldsymbol{I}-\boldsymbol{A})^{-1} \iff \rho(\boldsymbol{A})<1$ by Lemma 14 because $\boldsymbol{A}\nu=\lambda\nu \iff (\boldsymbol{I}-\boldsymbol{A})\nu=(1-\lambda)\nu$.

[(b) $\iff$ (c)]: By Lemma 13, $\rho(\boldsymbol{S}_{UU})<1$ and $\rho\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)<1 \iff \boldsymbol{W}_{UU} \succ 0$ and $\boldsymbol{\Delta}_{UU} \succ 0$. Either set of these equivalent conditions implies

$$\begin{aligned}
\boldsymbol{D}_{LL}(\boldsymbol{I}-\boldsymbol{A}) &= \boldsymbol{D}_{LL}\left(\boldsymbol{I}-\boldsymbol{S}_{LU}\boldsymbol{S}_{UU}^{-1}\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL}\right) \\
&= \boldsymbol{D}_{LL}\left(\boldsymbol{I}-\boldsymbol{S}_{LU}\boldsymbol{S}_{UU}^{-1}\boldsymbol{S}_{UL}-\boldsymbol{S}_{LU}\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL}\right) \\
&= \boldsymbol{D}_{LL}\left(\boldsymbol{I}-\boldsymbol{S}_{LU}\left(\boldsymbol{I}-\boldsymbol{S}_{UU}\right)^{-1}\boldsymbol{S}_{UL}-\boldsymbol{S}_{LL}\right)+\boldsymbol{D}_{LL}\left(\boldsymbol{S}_{LL}-\boldsymbol{S}_{LU}\boldsymbol{S}_{UU}^{-1}\boldsymbol{S}_{UL}\right) \qquad (39) \\
&= \left(\boldsymbol{\Delta}_{LL}-\boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}_{UU}^{-1}\boldsymbol{\Delta}_{UL}\right)+\left(\boldsymbol{W}_{LL}-\boldsymbol{W}_{LU}\boldsymbol{W}_{UU}^{-1}\boldsymbol{W}_{UL}\right) \\
&= \boldsymbol{W}_{LL}^{\star}+\boldsymbol{\Delta}_{LL}^{\star},
\end{aligned}$$

so $(\boldsymbol{W}^{\star}_{LL}+\boldsymbol{\Delta}^{\star}_{LL})^{-1}$ exists $\iff \rho(\boldsymbol{A})<1$.

[(c) $\iff$ (d)]: This follows automatically. ■

Proposition 16 *If* $\boldsymbol{W}\succeq 0$ *then*

$$(\boldsymbol{\Delta S})_{UU}\succ 0 \iff \boldsymbol{W}_{UU}\succ 0,\ \boldsymbol{\Delta}_{UU}\succ 0, \text{ and } (\boldsymbol{W}^{\star}_{LL}+\boldsymbol{\Delta}^{\star}_{LL})\succ 0.$$

Proof This is a special case of Lemma 15. ■

Lemma 17 *Let* $\boldsymbol{W}\succeq 0$*, and assume that* $\boldsymbol{\Gamma}_{LL}$ *exists. An equivalent form to that in Proposition 12 for the labeled solution to the joint training problem in Display (24) with* $\gamma=0$ *is* $f_L=\boldsymbol{\Gamma}_{LL}Y_L$.

Proof By Proposition 12 with $\gamma=0$, the joint training labeled estimator is

$$f_L=\left(\boldsymbol{S}_{LL}-\boldsymbol{S}_{LU}(\boldsymbol{\Delta S})^{-1}_{UU}(\boldsymbol{\Delta S})_{UL}\right)Y_L. \tag{40}$$

Now, it follows from some matrix algebra that

$$\begin{aligned}-(\boldsymbol{\Delta S})^{-1}_{UU}(\boldsymbol{\Delta S})_{UL} &= ((\boldsymbol{I}-\boldsymbol{S}_{UU})\boldsymbol{S}_{UU}-\boldsymbol{S}_{UL}\boldsymbol{S}_{LU})^{-1}(\boldsymbol{S}_{UL}\boldsymbol{S}_{LL}-(\boldsymbol{I}-\boldsymbol{S}_{UU})\boldsymbol{S}_{UL})\\ &= (\boldsymbol{I}-\boldsymbol{F})^{-1}(\boldsymbol{E}-\boldsymbol{S}^{-1}_{UU}\boldsymbol{S}_{UL}),\end{aligned} \tag{41}$$

where

$$\begin{aligned}\boldsymbol{E} &= \boldsymbol{S}^{-1}_{UU}(\boldsymbol{I}-\boldsymbol{S}_{UU})^{-1}\boldsymbol{S}_{UL}\boldsymbol{S}_{LL},\\ \boldsymbol{F} &= \boldsymbol{S}^{-1}_{UU}(\boldsymbol{I}-\boldsymbol{S}_{UU})^{-1}\boldsymbol{S}_{UL}\boldsymbol{S}_{LU}.\end{aligned}$$

Further simplification is based on an identity involving $\boldsymbol{A}$ from Lemma 14 and $\boldsymbol{F}$, that is,

$$\begin{aligned}\boldsymbol{S}_{LU}(\boldsymbol{I}-\boldsymbol{F})^{-1} &= \boldsymbol{S}_{LU}\left(\sum_{\ell=0}^{\infty}\left(\boldsymbol{S}^{-1}_{UU}(\boldsymbol{I}-\boldsymbol{S}_{UU})^{-1}\boldsymbol{S}_{UL}\boldsymbol{S}_{LU}\right)^{\ell}\right) &(42)\\ &= \left(\sum_{\ell=0}^{\infty}\left(\boldsymbol{S}_{LU}\boldsymbol{S}^{-1}_{UU}(\boldsymbol{I}-\boldsymbol{S}_{UU})^{-1}\boldsymbol{S}_{UL}\right)^{\ell}\right)\boldsymbol{S}_{LU} &(43)\\ &= (\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{S}_{LU}.\end{aligned}$$

The geometric matrix series in Display (43) converges because $\rho(\boldsymbol{A})<1$ by Lemma 15. Since $\boldsymbol{F}\nu=\lambda\nu \Longrightarrow \boldsymbol{A}\boldsymbol{S}_{LU}\nu=\lambda\boldsymbol{S}_{LU}\nu$ and $\nu^T\boldsymbol{A}=\lambda\nu^T \Longrightarrow \nu^T\boldsymbol{S}_{LU}\boldsymbol{F}=\lambda\nu^T\boldsymbol{S}_{LU}$, $\boldsymbol{F}$ and $\boldsymbol{A}$ have the same non-zero eigenvalues, so the infinite series in Display (42) is also well-defined.

Substitutions of Display (41) and $\boldsymbol{S}_{LU}(\boldsymbol{I}-\boldsymbol{F})^{-1}=(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{S}_{LU}$ produce

$$\begin{aligned}\boldsymbol{S}_{LL}-\boldsymbol{S}_{LU}(\boldsymbol{\Delta S})^{-1}_{UU}(\boldsymbol{\Delta S})_{UL} &= \boldsymbol{S}_{LL}+\boldsymbol{S}_{LU}(\boldsymbol{I}-\boldsymbol{F})^{-1}(\boldsymbol{E}-\boldsymbol{S}^{-1}_{UU}\boldsymbol{S}_{UL})\\ &= \boldsymbol{S}_{LL}+(\boldsymbol{I}-\boldsymbol{A})^{-1}(\boldsymbol{S}_{LU}\boldsymbol{E}-\boldsymbol{S}_{LU}\boldsymbol{S}^{-1}_{UU}\boldsymbol{S}_{UL})\\ &= \boldsymbol{S}_{LL}+(\boldsymbol{I}-\boldsymbol{A})^{-1}(\boldsymbol{A}\boldsymbol{S}_{LL}-\boldsymbol{S}_{LU}\boldsymbol{S}^{-1}_{UU}\boldsymbol{S}_{UL})\\ &= \left(\boldsymbol{I}+(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{A}\right)\boldsymbol{S}_{LL}-(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{S}_{LU}\boldsymbol{S}^{-1}_{UU}\boldsymbol{S}_{UL}\\ &= (\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{S}^{\star}_{LL}.\end{aligned}$$

Therefore, the equivalent form $f_L = \mathbf{\Gamma}_{LL} Y_L = (\boldsymbol{W}^\star_{LL} + \boldsymbol{\Delta}^\star_{LL})^{-1} \boldsymbol{W}^\star_{LL} Y_L$ for Equation (40) is established using $\boldsymbol{D}_{LL}(\boldsymbol{I} - \boldsymbol{A}) = \boldsymbol{W}^\star_{LL} + \boldsymbol{\Delta}^\star_{LL}$ from Display (39) and $\boldsymbol{S}^\star_{LL} = \boldsymbol{D}^{-1}_{LL} \boldsymbol{W}^\star_{LL}$. ■

Theorem 18 *Let $\boldsymbol{W} \succeq 0$, and assume that $\mathbf{\Gamma}_{LL}$ exists. The solution to the Joint Harmonic Optimization Problem (24) with $\gamma = 0$ has*

$$f = \begin{pmatrix} f_L \\ (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} f_L \end{pmatrix} = \begin{pmatrix} \mathbf{\Gamma}_{LL} \\ (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} \mathbf{\Gamma}_{LL} \end{pmatrix} Y_L,$$

so f is in-fact harmonic.

Proof The optimal $\hat{Y}_U$ satisfies the derivative score in Display (34) with $\gamma = 0$, so

$$\hat{Y}_U = f_U - \boldsymbol{S}^{-1}_{UU} \boldsymbol{S}_{UL} (Y_L - f_L)$$

after rearrangement. Finally, since the optimal f satisfies $f = \boldsymbol{S} Y(\hat{Y}_U)$, f_U satisfies

$$\begin{aligned} f_U &= \boldsymbol{S}_{UL} Y_L + \boldsymbol{S}_{UU} \hat{Y}_U \\ &= \boldsymbol{S}_{UL} Y_L + \boldsymbol{S}_{UU} f_U - \boldsymbol{S}_{UL} (Y_L - f_L) \\ &= (\boldsymbol{I} - \boldsymbol{S}_{UU})^{-1} \boldsymbol{S}_{UL} f_L, \end{aligned}$$

and the optimal f_L satisfies $f_L = \mathbf{\Gamma}_{LL} Y_L$ by Lemma 17. ■

Proposition 19 *If $\boldsymbol{W} \succeq 0$ and $\mathbf{\Gamma}_{LL}$ exists then each eigenvalue of $\mathbf{\Gamma}_{LL}$ is an element of $[0, 1]$.*

Proof Since $\boldsymbol{W}_{UU} \succ 0$ by Lemma 15, $\boldsymbol{W} \succeq 0 \iff \boldsymbol{W}^\star_{LL} \succeq 0$, so it is well-defined to set

$$\boldsymbol{V}_3 = \begin{pmatrix} \boldsymbol{I} & {\boldsymbol{W}^\star_{LL}}^{1/2} \\ {\boldsymbol{W}^\star_{LL}}^{1/2} & \boldsymbol{W}^\star_{LL} + \boldsymbol{\Delta}^\star_{LL} \end{pmatrix}.$$

The $\boldsymbol{I}$ block Schur complement of $\boldsymbol{V}_3$ is $\boldsymbol{\Delta}^\star_{LL} \succeq 0$, so the other block is positive semi-definite, that is,

$$\boldsymbol{I} - {\boldsymbol{W}^\star_{LL}}^{1/2} (\boldsymbol{W}^\star_{LL} + \boldsymbol{\Delta}^\star_{LL})^{-1} {\boldsymbol{W}^\star_{LL}}^{1/2} \succeq 0,$$

and $\mathbf{\Gamma}_{LL}$ and ${\boldsymbol{W}^\star_{LL}}^{1/2} (\boldsymbol{W}^\star_{LL} + \boldsymbol{\Delta}^\star_{LL})^{-1} {\boldsymbol{W}^\star_{LL}}^{1/2} \succeq 0$ have the same eigenvalues. ■

A.4 Regularized Joint Harmonic Estimators $\gamma > 0$

Lemma 20 *Let $\boldsymbol{W} \succeq 0$ and $\gamma > 0$ and define*

$$\mathbf{\Gamma}_{LL_\gamma} = \left(\boldsymbol{W}^\star_{LL_\gamma} + \boldsymbol{\Delta}^\star_{LL_\gamma} \right)^{-1} \boldsymbol{W}^\star_{LL_\gamma}.$$

The labeled solution to the Joint Optimization Problem (24) is equivalently given by $f_{L_\gamma} = \mathbf{\Gamma}_{LL_\gamma} Y_L$.

Proof The sum of "regularized inverses" in Displays (29) and (30)

$$\boldsymbol{C}_\gamma = \boldsymbol{W}_{\overline{UU}_\gamma} + \boldsymbol{\Delta}^-_{\overline{UU}_\gamma} = (\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UU} + \gamma\boldsymbol{I})^{-1}$$

is positive definite by Proposition 11, and

$$((\boldsymbol{\Delta S})_{UU} + \gamma\boldsymbol{I})^{-1}(\boldsymbol{\Delta S})_{UL} \;=\; \boldsymbol{G}_\gamma + \boldsymbol{H}_\gamma, \tag{44}$$

where

$$\begin{aligned}\boldsymbol{G}_\gamma &= \left(\boldsymbol{I} - \boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\right)^{-1}\boldsymbol{C}_\gamma\boldsymbol{\Delta}_{UL}\boldsymbol{S}_{LL},\\ \boldsymbol{H}_\gamma &= \left(\boldsymbol{I} - \boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\right)^{-1}\boldsymbol{C}_\gamma\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UL}.\end{aligned}$$

Thus, by Proposition 12, labeled estimator f_L depends on

$$\boldsymbol{S}_{LL} - \boldsymbol{S}_{LU}((\boldsymbol{\Delta S})_{UU} + \gamma\boldsymbol{I})^{-1}(\boldsymbol{\Delta S})_{UL} = \boldsymbol{S}_{LL} - \boldsymbol{S}_{LU}\boldsymbol{G}_\gamma - \boldsymbol{S}_{LU}\boldsymbol{H}_\gamma. \tag{45}$$

Simplification of terms on the right of Equation (45) is based on

$$\begin{aligned}\left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{D}_{LL}^{-1} &= \left(\boldsymbol{D}_{LL} - \boldsymbol{W}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\\ &= \left(\boldsymbol{D}_{LL} - \boldsymbol{\Delta}_{LU}\boldsymbol{\Delta}^-_{\overline{UU}_\gamma}\boldsymbol{\Delta}_{UL} - \boldsymbol{W}_{LU}\boldsymbol{W}^-_{\overline{UU}_\gamma}\boldsymbol{W}_{UL}\right)^{-1}\\ &= \left(\boldsymbol{W}^\star_{\overline{LL}_\gamma} + \boldsymbol{\Delta}^\star_{\overline{LL}_\gamma}\right)^{-1}\end{aligned}$$

and on

$$\boldsymbol{S}_{LU}\left(\boldsymbol{I} - \boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\right)^{-1} = \left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{S}_{LU}$$

if $\rho\left(\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right) < 1$ by a geometric matrix series argument similar to that used to establish Displays (42) and (43). Because $\gamma > 0$ is shrinking the eigenvalues of $\boldsymbol{C}_\gamma$, $\rho\left(\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right) < 1$ as a consequence of a generalization of Lemma 14 since $\boldsymbol{B}_1$ is unique even if arbitrary generalized inverses are used to compute the Schur complements in Display (37). Now, terms on the right of Equation (45) reduce to

$$\begin{aligned}\boldsymbol{S}_{LL} - \boldsymbol{S}_{LU}\boldsymbol{G}_\gamma &= \left(\boldsymbol{I} + \boldsymbol{S}_{LU}\left(\boldsymbol{I} - \boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\right)^{-1}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)\boldsymbol{S}_{LL}\\ &= \left(\boldsymbol{I} + \left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)\boldsymbol{S}_{LL}\\ &= \left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{D}_{LL}^{-1}\boldsymbol{W}_{LL}\\ &= \left(\boldsymbol{W}^\star_{\overline{LL}_\gamma} + \boldsymbol{\Delta}^\star_{\overline{LL}_\gamma}\right)^{-1}\boldsymbol{W}_{LL}\end{aligned} \tag{46}$$

and

$$\begin{aligned}\boldsymbol{S}_{LU}\boldsymbol{H}_\gamma &= \boldsymbol{S}_{LU}\left(\boldsymbol{I} - \boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\right)^{-1}\boldsymbol{C}_\gamma\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UL}\\ &= \left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma(\boldsymbol{I} - \boldsymbol{S}_{UU})^T\boldsymbol{W}_{UL}\\ &= \left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{D}_{LL}^{-1}\left(\boldsymbol{W}_{LU}\boldsymbol{W}^-_{\overline{UU}_\gamma}\boldsymbol{W}_{UL}\right)\\ &= \left(\boldsymbol{W}^\star_{\overline{LL}_\gamma} + \boldsymbol{\Delta}^\star_{\overline{LL}_\gamma}\right)^{-1}\boldsymbol{W}_{LU}\boldsymbol{W}^-_{\overline{UU}_\gamma}\boldsymbol{W}_{UL}.\end{aligned} \tag{47}$$

The right of Equation (45) simplifies to $\boldsymbol{\Gamma}_{\overline{LL}_\gamma}$ based on Equations (46) and (47). ■

Theorem 21 *Let $\boldsymbol{W} \succeq 0$. Let f_γ denote the solution to the Joint Harmonic Optimization Problem (24) with $\gamma > 0$. Then*

$$f_\gamma = \begin{pmatrix} \boldsymbol{\Gamma}_{LL_\gamma} \\ -\left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \boldsymbol{\Delta}_{UL}\boldsymbol{\Gamma}_{LL_\gamma} + \left(\boldsymbol{I} - \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \boldsymbol{\Delta}_{UU}\right)\boldsymbol{S}_{UL} \end{pmatrix} Y_L.$$

Proof Matrix definitions and techniques from the proof of Lemma 20 are used here. Let

$$\begin{aligned} \boldsymbol{R}_\gamma &= \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \boldsymbol{W}_{UL}\left(\boldsymbol{W}_{LL_\gamma}^\star + \boldsymbol{\Delta}_{LL_\gamma}^\star\right)^{-1} \boldsymbol{W}_{LU}\boldsymbol{W}_{UU_\gamma}^- \boldsymbol{W}_{UL} \\ &= \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \left\{\boldsymbol{W}_{UL}\left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\right\}\left(\boldsymbol{I} - \boldsymbol{S}_{UU}\right)^T \boldsymbol{W}_{UL} \\ &= \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \left\{\left(\boldsymbol{I} - \boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\right)^{-1}\boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\right\}\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UL}. \end{aligned} \tag{48}$$

Then

$$\begin{aligned} \boldsymbol{S}_{UU}\boldsymbol{G}_\gamma &= \boldsymbol{S}_{UU}\left(\boldsymbol{I} - \boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\right)^{-1}\boldsymbol{C}_\gamma\boldsymbol{\Delta}_{UL}\boldsymbol{S}_{LL} \\ &= \boldsymbol{S}_{UU}\boldsymbol{C}_\gamma\boldsymbol{\Delta}_{UL}\left(\boldsymbol{I} - \boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\boldsymbol{W}_{UL}\right)^{-1}\boldsymbol{S}_{LL} \\ &= \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \boldsymbol{\Delta}_{UL}\left(\boldsymbol{W}_{LL_\gamma}^\star + \boldsymbol{\Delta}_{LL_\gamma}^\star\right)^{-1}\boldsymbol{W}_{LL} \\ &= \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \boldsymbol{\Delta}_{UL}\left(\boldsymbol{W}_{LL_\gamma}^\star + \boldsymbol{\Delta}_{LL_\gamma}^\star\right)^{-1}\boldsymbol{W}_{LL}^\star + \boldsymbol{R}_\gamma \\ &= \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \boldsymbol{\Delta}_{UL}\boldsymbol{\Gamma}_{LL_\gamma} + \boldsymbol{R}_\gamma. \end{aligned} \tag{49}$$

Equation (48) and $\boldsymbol{S}_{UU}\boldsymbol{H}_\gamma = \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \left\{\left(\boldsymbol{I} - \boldsymbol{W}_{UL}\boldsymbol{S}_{LU}\boldsymbol{C}_\gamma\right)^{-1}\right\}\boldsymbol{\Delta}_{UU}\boldsymbol{S}_{UL}$ imply

$$\boldsymbol{S}_{UL} - \left(\boldsymbol{S}_{UU}\boldsymbol{H}_\gamma + \boldsymbol{R}_\gamma\right) = \left(\boldsymbol{I} - \left(\boldsymbol{\Delta}_{UU_\gamma}^-\right)^T \{\boldsymbol{I}\}\,\boldsymbol{\Delta}_{UU}\right)\boldsymbol{S}_{UL}. \tag{50}$$

Proposition 12 and Equation (44) result in the unlabeled estimator smoother

$$\boldsymbol{S}_{UL} - \boldsymbol{S}_{UU}(\boldsymbol{G}_\gamma + \boldsymbol{H}_\gamma) = \boldsymbol{S}_{UL} - \left(\boldsymbol{S}_{UU}\boldsymbol{H}_\gamma + \boldsymbol{R}_\gamma\right) - (\boldsymbol{S}_{UU}\boldsymbol{G}_\gamma - \boldsymbol{R}_\gamma), \tag{51}$$

and substitutions based on Equations (49) and (50) into the right of Equation (51) produce its desired form. The labeled estimator smoother $\boldsymbol{\Gamma}_{LL_\gamma}$ is given by Lemma 20. ■

References

S Abney. *Semisupervised Learning for Computational Linguistics*. Chapman and Hall, CRC, 2008.

A Aswani, P Bickel, and C Tomlin. Regression on manifolds: estimation of the exterior derivative. *Annals of Statistics*, 39(1):48–81, 2010.

M Azizyan, A Singh, and L Wasserman. Density-sensitive semisupervised inference. *Annals of Statistics*, 41(2):751–771, 2013.

M Belkin, P Niyogi, and V Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

M Bredel and E Jacoby. Chemogenomics: An emerging strategy for rapid target and drug discovery. *Nature Reviews Genetics*, 5(4):262–275, April 2004.

M Carreira-Perpiñán and R Zemel. Proximity graphs for clustering and manifold learning. In *Advances in NIPs 18*, pages 225–232, 2005.

O Chapelle, M Chi, and A Zien. A continuation method for semi-supervised SVMs. In *International Conference on Machine Learning*, 2006a.

O Chapelle, B Schölkopf, and A Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006b. URL `http://www.kyb.tuebingen.mpg.de/ssl-book`.

C Cortes, M Mohri, D Pechyony, and A Rastogi. Stability of transductive regression algorithms. In *International Conference of Machine Learning*, 2008.

M Culp. On the semi-supervised joint trained elastic net. *Journal of Computational Graphics and Statistics*, 22(2):300–318, 2013.

M Culp, G Michailidis, and K Johnson. On multi-view learning with additive models. *Annals of Applied Statistics*, 3(1):545–571, 2009.

P Doyle and J Snell. Random walks and electrical networks. *Mathematical Association of America*, 1984.

A Frank and A Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

T Hastie, R Tibshirani, and J Friedman. *The Elements of Statistical Learning (Data Mining, Inference and Prediction)*. Springer Verlag, 2001.

M Hein, J Audibert, and U von Luxburg. From graphs to manifolds–weak and strong pointwise consistency of graph Laplacians. In *Conference on Learning Theory*, pages 470–485, 2005.

A Izenman. *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning*. Springer Verlag, 2008.

T Jebara, J Wang, and S Chang. Graph construction and b-matching for semi-supervised learning. In *International Conference of Machine Learning*, 2009.

I Koprinska, J Poon, J Clark, and J Chan. Learning to classify e-mail. *Information Science*, 177 (10):2167–2187, 2007. ISSN 0020-0255.

M Kui, K Zhang, S Mehta, T Chen, and F Sun. Prediction of protein function using protein-protein interaction data. *Journal of Computational Biology*, 10:947–960, 2002.

J Lafferty and L Wasserman. Statistical analysis of semi-supervised regression. In *Advances in NIPS*, pages 801–808. MIT Press, 2007.

R Lundblad. *Chemical Reagents for Protein Modification*. CRC Press Inc., 2004. ISBN 08493-1983-8.

A McCallum, K Nigam, J Rennie, and K Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval Journal*, 3:127–163, 2000.

D Meyer, E Dimitriadou, K Hornik, A Weingessel, and F Leisch. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2012. URL `http://CRAN.R-project.org/package=e1071`. R package version 1.6-1.

B Nadler, N Srebro, and X Zhou. Statistical analysis of semi-supervised learning: The limit of infinite unlabelled data. In *Advances in NIPs 22*, pages 1330–1338. MIT Press, 2009.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2012.

P Rigollet. Generalization error bounds in semi-supervised classification under the cluster assumption. *Journal of Machine Learning Research*, 8:1369–1392, 2007.

A Singh, R Nowak, and X Zhu. Unlabeled data: Now it helps, now it doesn't. In *Advanced in NIPS*, pages 1513–1520, 2008.

A Subramanya and J Bilmes. Semi-supervised learning with measure propagation. *Journal of Machine Learning Research*, 12:3311–3370, 2011.

U von Luxburg, A Radl, and M Hein. Hitting times, commute distances and the spectral gap for large random geometric graphs. *Computing Research Repository*, abs/1003.1266, 2010.

J Wang and X Shen. Large margin semi-supervised learning. *Journal of Machine Learning Research*, 8:1867–1897, 2007.

Y Yamanishi, J Vert, and M Kanehisa. Protein network inference from multiple genomic data: A supervised approach. *Bioinformatics*, 20:363–370, 2004.

X Zhu. Semi-supervised learning literature survey. Technical report, Computer Sciences, University of Wisconsin-Madison, 2008.

X Zhu and A Goldberg. *Introduction to Semi-Supervised Learning*. Morgan and Claypool Publishers, 2009.

X Zhu, Z Ghahramani, and J Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *International Conference on Machine Learning*, pages 912–919, 2003.

Journal of Machine Learning Research 14 (2013) 3753-3783 Submitted 12/11; Revised 6/13; Published 12/13

Kernel Bayes' Rule: Bayesian Inference with Positive Definite Kernels

Kenji Fukumizu FUKUMIZU@ISM.AC.JP
The Institute of Statistical Mathematics
10-3 Midoricho, Tachikawa
Tokyo 190-8562 Japan

Le Song LSONG@CC.GATECH.EDU
College of Computing
Georgia Institute of Technology
1340 Klaus Building, 266 Ferst Drive
Atlanta, GA 30332, USA

Arthur Gretton ARTHUR.GRETTON@GMAIL.COM
Gatsby Computational Neuroscience Unit
University College London
Alexandra House, 17 Queen Square
London, WC1N 3AR, UK

Editor: Ingo Steinwart

Abstract

A kernel method for realizing Bayes' rule is proposed, based on representations of probabilities in reproducing kernel Hilbert spaces. Probabilities are uniquely characterized by the mean of the canonical map to the RKHS. The prior and conditional probabilities are expressed in terms of RKHS functions of an empirical sample: no explicit parametric model is needed for these quantities. The posterior is likewise an RKHS mean of a weighted sample. The estimator for the expectation of a function of the posterior is derived, and rates of consistency are shown. Some representative applications of the kernel Bayes' rule are presented, including Bayesian computation without likelihood and filtering with a nonparametric state-space model.

Keywords: kernel method, Bayes' rule, reproducing kernel Hilbert space

1. Introduction

Kernel methods have long provided powerful tools for generalizing linear statistical approaches to nonlinear settings, through an embedding of the sample to a high dimensional feature space, namely a reproducing kernel Hilbert space (RKHS) (Schölkopf and Smola, 2002). Examples include support vector machines, kernel PCA, and kernel CCA, among others. In these cases, data are mapped via a canonical feature map to a reproducing kernel Hilbert space (of high or even infinite dimension), in which the linear operations that define the algorithms are implemented. The inner product between feature mappings need never be computed explicitly, but is given by a positive definite kernel function unique to the RKHS: this permits efficient computation without the need to deal explicitly with the feature representation.

The mappings of individual points to a feature space may be generalized to mappings of probability measures (e.g., Berlinet and Thomas-Agnan, 2004, Chapter 4). We call such mappings the

kernel means of the underlying random variables. With an appropriate choice of positive definite kernel, the kernel mean on the RKHS uniquely determines the distribution of the variable (Fukumizu et al., 2004, 2009a; Sriperumbudur et al., 2010), and statistical inference problems on distributions can be solved via operations on the kernel means. Applications of this approach include homogeneity testing (Gretton et al., 2007; Harchaoui et al., 2008; Gretton et al., 2009a, 2012), where the empirical means on the RKHS are compared directly, and independence testing (Gretton et al., 2008, 2009b), where the mean of the joint distribution on the feature space is compared with that of the product of the marginals. Representations of conditional dependence may also be defined in RKHS, and have been used in conditional independence tests (Fukumizu et al., 2008; Zhang et al., 2011).

In this paper, we propose a novel, nonparametric approach to Bayesian inference, making use of kernel means of probabilities. In applying Bayes' rule, we compute the posterior probability of x in $\mathcal{X}$ given observation y in $\mathcal{Y}$;

$$q(x|y) = \frac{p(y|x)\pi(x)}{q_{\mathcal{Y}}(y)}, \tag{1}$$

where $\pi(x)$ and $p(y|x)$ are the density functions of the prior and the likelihood of y given x, respectively, with respective base measures $\nu_{\mathcal{X}}$ and $\nu_{\mathcal{Y}}$, and the normalization factor $q_{\mathcal{Y}}(\mathrm{y})$ is given by

$$q_{\mathcal{Y}}(y) = \int p(y|x)\pi(x)d\nu_{\mathcal{X}}(x).$$

Our main result is a nonparametric estimate of posterior kernel mean, given kernel mean representations of the prior and likelihood. We call this method *kernel Bayes' rule*.

A valuable property of the kernel Bayes' rule is that the kernel posterior mean is estimated nonparametrically from data. The prior is represented by a weighted sum over a sample, and the probabilistic relation expressed by the likelihood is represented in terms of a sample from a joint distribution having the desired conditional probability. This confers an important benefit: we can still perform Bayesian inference by making sufficient observations on the system, even in the absence of a specific parametric model of the relation between variables. More generally, if we can sample from the model, we do not require explicit density functions for inference. Such situations are typically seen when the prior or likelihood is given by a random process: Approximate Bayesian Computation (Tavaré et al., 1997; Marjoram et al., 2003; Sisson et al., 2007) is widely applied in population genetics, where the likelihood is expressed as a branching process, and nonparametric Bayesian inference (Müller and Quintana, 2004) often uses a process prior with sampling methods. Alternatively, a parametric model may be known, however it might be of sufficient complexity to require Markov chain Monte Carlo or sequential Monte Carlo for inference. The present kernel approach provides an alternative strategy for Bayesian inference in these settings. We demonstrate consistency for our posterior kernel mean estimate, and derive convergence rates for the expectation of functions computed using this estimate.

An alternative to the kernel mean representation would be to use nonparametric density estimates for the posterior. Classical approaches include kernel density estimation (KDE) or distribution estimation on a finite partition of the domain. These methods are known to perform poorly on high dimensional data, however. In addition, computation of the posterior with KDE requires importance weights, which may not be accurate in low density areas. By contrast, the proposed kernel mean representation is defined as an integral or moment of the distribution, taking the form of a function in an RKHS. Thus, it is more akin to the characteristic function approach (see, e.g.,

Kankainen and Ushakov, 1998) to representing probabilities. A well conditioned empirical estimate of the characteristic function can be difficult to obtain, especially for conditional probabilities. By contrast, the kernel mean has a straightforward empirical estimate, and conditioning and marginalization can be implemented easily, at a reasonable computational cost.

The proposed method of realizing Bayes' rule is an extension of the approach used by Song et al. (2009) for state-space models. In this earlier work, a heuristic approximation was used, where the kernel mean of the new hidden state was estimated by adding kernel mean estimates from the previous hidden state and the observation. Another relevant work is the belief propagation approach in Song et al. (2010a, 2011), which covers the simpler case of a uniform prior.

This paper is organized as follows. We begin in Section 2 with a review of RKHS terminology and of kernel mean embeddings. In Section 3, we derive an expression for Bayes' rule in terms of kernel means, and provide consistency guarantees. We apply the kernel Bayes' rule in Section 4 to various inference problems, with numerical results and comparisons with existing methods in Section 5. Our proofs are contained in Section 6 (including proofs of the consistency results of Section 3).

2. Preliminaries: Positive Definite Kernels and Probabilities

Throughout this paper, all Hilbert spaces are assumed to be separable. For an operator A on a Hilbert space, the range is denoted by $\mathcal{R}(A)$. The linear hull of a subset S in a vector space is denoted by Span$S$.

We begin with a review of positive definite kernels, and of statistics on the associated reproducing kernel Hilbert spaces (Aronszajn, 1950; Berlinet and Thomas-Agnan, 2004; Fukumizu et al., 2004, 2009a). Given a set Ω, a ($\mathbb{R}$-valued) positive definite kernel k on Ω is a symmetric kernel $k : \Omega \times \Omega \to \mathbb{R}$ such that $\sum_{i,j=1}^{n} c_i c_j k(x_i, x_j) \geq 0$ for arbitrary number of points $x_1, \ldots, x_n$ in Ω and real numbers $c_1, \ldots, c_n$. The matrix $(k(x_i, x_j))_{i,j=1}^{n}$ is called a Gram matrix. It is known by the Moore-Aronszajn theorem (Aronszajn, 1950) that a positive definite kernel on Ω uniquely defines a Hilbert space $\mathcal{H}$ consisting of functions on Ω such that the following three conditions hold:

(i) $k(\cdot, x) \in \mathcal{H}$ for any $x \in \Omega$,

(ii) Span$\{k(\cdot, x) \mid x \in \Omega\}$ is dense in $\mathcal{H}$,

(iii) $\langle f, k(\cdot, x) \rangle = f(x)$ for any $x \in \Omega$ and $f \in \mathcal{H}$ (the reproducing property), where $\langle \cdot, \cdot \rangle$ is the inner product of $\mathcal{H}$.

The Hilbert space $\mathcal{H}$ is called the *reproducing kernel Hilbert space* (RKHS) associated with k, since the function $k_x = k(\cdot, x)$ serves as the reproducing kernel $\langle f, k_x \rangle = f(x)$ for $f \in \mathcal{H}$.

A positive definite kernel on Ω is said to be *bounded* if there is $M > 0$ such that $k(x, x) \leq M$ for any $x \in \Omega$.

Let $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ be a measurable space, X be a random variable taking values in $\mathcal{X}$ with distribution $P_\mathcal{X}$, and k be a measurable positive definite kernel on $\mathcal{X}$ such that $E[\sqrt{k(X, X)}] < \infty$. The associated RKHS is denoted by $\mathcal{H}$. The *kernel mean* m_X^k (also written $m_{P_X}^k$) of X on the RKHS $\mathcal{H}$ is defined by the mean of the $\mathcal{H}$-valued random variable $k(\cdot, X)$. The existence of the kernel mean is guaranteed by $E[\|k(\cdot, X)\|] = E[\sqrt{k(X, X)}] < \infty$. We will generally write m_X in place of m_X^k for simplicity, where there is no ambiguity. By the reproducing property, the kernel mean satisfies the relation

$$\langle f, m_X \rangle = E[f(X)] \tag{2}$$

for any $f \in \mathcal{H}$. Plugging $f = k(\cdot, u)$ into this relation,

$$m_X(u) = E[k(u,X)] = \int k(u,\tilde{x}) dP_X(\tilde{x}), \tag{3}$$

which shows the explicit functional form. The kernel mean m_X is also denoted by m_{P_X}, as it depends only on the distribution P_X with k fixed.

Let $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ and $(\mathcal{Y}, \mathcal{B}_\mathcal{Y})$ be measurable spaces, (X,Y) be a random variable on $\mathcal{X} \times \mathcal{Y}$ with distribution P, and $k_\mathcal{X}$ and $k_\mathcal{Y}$ be measurable positive definite kernels with respective RKHS $\mathcal{H}_\mathcal{X}$ and $\mathcal{H}_\mathcal{Y}$ such that $E[k_\mathcal{X}(X,X)] < \infty$ and $E[k_\mathcal{Y}(Y,Y)] < \infty$. The (uncentered) *covariance operator* $C_{YX} : \mathcal{H}_\mathcal{X} \to \mathcal{H}_\mathcal{Y}$ is defined as the linear operator that satisfies

$$\langle g, C_{YX} f \rangle_{\mathcal{H}_\mathcal{Y}} = E[f(X)g(Y)]$$

for all $f \in \mathcal{H}_\mathcal{X}, g \in \mathcal{H}_\mathcal{Y}$. This operator C_{YX} can be identified with $m_{(YX)}$ in the product space $\mathcal{H}_\mathcal{Y} \otimes \mathcal{H}_\mathcal{X}$, which is given by the product kernel $k_\mathcal{Y} k_\mathcal{X}$ on $\mathcal{Y} \times \mathcal{X}$ (Aronszajn, 1950), by the standard identification between the linear maps and the tensor product. We also define C_{XX} for the operator on $\mathcal{H}_\mathcal{X}$ that satisfies $\langle f_2, C_{XX} f_1 \rangle = E[f_2(X) f_1(X)]$ for any $f_1, f_2 \in \mathcal{H}_\mathcal{X}$. Similarly to Equation (3), the explicit integral expressions for C_{YX} and C_{XX} are given by

$$(C_{YX} f)(y) = \int k_\mathcal{Y}(y,\tilde{y}) f(\tilde{x}) dP(\tilde{x},\tilde{y}) \quad \text{and} \quad (C_{XX} f)(x) = \int k_\mathcal{X}(x,\tilde{x}) f(\tilde{x}) dP_\mathcal{X}(\tilde{x}), \tag{4}$$

respectively.

An important notion in statistical inference with positive definite kernels is the characteristic property. A bounded measurable positive definite kernel k on a measurable space $(\Omega, \mathcal{B})$ is called *characteristic* if the mapping from a probability Q on $(\Omega, \mathcal{B})$ to the kernel mean $m_Q^k \in \mathcal{H}$ is injective (Fukumizu et al., 2009a; Sriperumbudur et al., 2010). This is equivalent to assuming that $E_{X\sim P}[k(\cdot,X)] = E_{X'\sim Q}[k(\cdot,X')]$ implies $P = Q$: probabilities are uniquely determined by their kernel means on the associated RKHS. With this property, problems of statistical inference can be cast as inference on the kernel means. A popular example of a characteristic kernel defined on Euclidean space is the Gaussian RBF kernel $k(x,y) = \exp(-\|x-y\|^2/(2\sigma^2))$. A bounded measurable positive definite kernel on a measurable space $(\Omega, \mathcal{B})$ with corresponding RKHS $\mathcal{H}$ is characteristic if and only if $\mathcal{H} + \mathbb{R}$ is dense in $L^2(P)$ for arbitrary probability P on $(\Omega, \mathcal{B})$, where $\mathcal{H} + \mathbb{R}$ is the direct sum of two RKHSs $\mathcal{H}$ and $\mathbb{R}$ (Aronszajn, 1950). This implies that the RKHS defined by a characteristic kernel is rich enough to be dense in L^2 space up to the constant functions. Other useful conditions for a kernel to be characteristic can be found in Sriperumbudur et al. (2010), Fukumizu et al. (2009b), and Sriperumbudur et al. (2011).

Throughout this paper, when positive definite kernels on a measurable space are discussed, the following assumption is made:

(K) Positive definite kernels are bounded and measurable.

Under this assumption, the mean and covariance always exist for arbitrary probabilities.

Given i.i.d. sample $(X_1,Y_1), \ldots, (X_n,Y_n)$ with law P, the empirical estimators of the kernel mean and covariance operator are given straightforwardly by

$$\widehat{m}_X^{(n)} = \frac{1}{n} \sum_{i=1}^n k_\mathcal{X}(\cdot, X_i), \qquad \widehat{C}_{YX}^{(n)} = \frac{1}{n} \sum_{i=1}^n k_\mathcal{Y}(\cdot, Y_i) \otimes k_\mathcal{X}(\cdot, X_i),$$

where $\widehat{C}_{YX}^{(n)}$ is written in tensor form. These estimators are $\sqrt{n}$-consistent in appropriate norms, and $\sqrt{n}(\widehat{m}_X^{(n)} - m_X)$ converges to a Gaussian process on $\mathcal{H}_\mathcal{X}$ (Berlinet and Thomas-Agnan, 2004, Section 9.1). While we may use non-i.i.d. samples for numerical examples in Section 5, in our theoretical analysis we always assume i.i.d. samples for simplicity.

3. Kernel Expression of Bayes' Rule

We review Bayes' rule and the notion of kernel conditional mean embeddings in Section 3.1. We demonstrate that Bayes' rule may be expressed in terms of these conditional mean embeddings. We provide consistency results for the empirical estimators of the conditional mean embedding for the posterior in Section 3.2.

3.1 Kernel Bayes' Rule

We first review Bayes' rule in a general form without using density functions, since the kernel Bayes' rule can be applied to situations where density functions are not available.

Let $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ and $(\mathcal{Y}, \mathcal{B}_\mathcal{Y})$ be measurable spaces, $(\Omega, \mathcal{A}, \mathbb{P})$ a probability space, and $(X,Y) : \Omega \to \mathcal{X} \times \mathcal{Y}$ be a $(\mathcal{X} \times \mathcal{Y}$-valued) random variable with distribution P. The marginal distribution of X is denoted by $P_\mathcal{X}$. Suppose that Π is a probability measure on $(\mathcal{X}, \mathcal{B}_\mathcal{X})$, which serves as a *prior* distribution. For each $x \in \mathcal{X}$, let $P_{\mathcal{Y}|x}$ denote the conditional probability of Y given $X = x$; namely, $P_{\mathcal{Y}|x}(B) = E[I_B(Y)|X = x]$, where I_B is the indicator function of a measurable set $B \in \mathcal{B}_\mathcal{Y}$.[1] We assume that the conditional probability $P_{\mathcal{Y}|x}$ is *regular*; namely, it defines a probability measure on $\mathcal{Y}$ for each x. The prior Π and the family $\{P_{\mathcal{Y}|x} \mid x \in \mathcal{X}\}$ defines the joint distribution Q on $\mathcal{X} \times \mathcal{Y}$ by

$$Q(A \times B) = \int_A P_{\mathcal{Y}|x}(B) d\Pi(x) \tag{5}$$

for any $A \in \mathcal{B}_\mathcal{X}$ and $B \in \mathcal{B}_\mathcal{Y}$, and its marginal distribution $Q_\mathcal{Y}$ by

$$Q_\mathcal{Y}(B) = Q(\mathcal{X} \times B).$$

Let (Z,W) be a random variable on $\mathcal{X} \times \mathcal{Y}$ with distribution Q. For $y \in \mathcal{Y}$, the *posterior* probability given y is defined by the conditional probability

$$Q_{\mathcal{X}|y}(A) = E[I_A(Z)|W = y] \qquad (A \in \mathcal{B}_\mathcal{X}). \tag{6}$$

If the probability distributions have density functions with respect to a measure $\nu_\mathcal{X}$ on $\mathcal{X}$ and $\nu_\mathcal{Y}$ on $\mathcal{Y}$, namely, if the p.d.f. of P and Π are given by $p(x,y)$ and $\pi(x)$, respectively, Equations (5) and (6) are reduced to the well known form Equation (1). To make Bayesian inference meaningful, we make the following assumption:

(A) The prior Π is absolutely continuous with respect to the marginal distribution $P_\mathcal{X}$.

1. The $\mathcal{B}_\mathcal{X}$-measurable function $P_{\mathcal{Y}|x}(B)$ is always well defined. In fact, the finite measure $\mu(A) := \int_{\{\omega \in \Omega | X(\omega) \in A\}} I_B(Y) dP$ on $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ is absolutely continuous with respect to $P_\mathcal{X}$. The Radon-Nikodym theorem then guarantees the existence of a $\mathcal{B}_\mathcal{X}$-measurable function $\eta(x)$ such that $\int_{\{\omega \in \Omega | X(\omega) \in A\}} I_B(Y) dP = \int_A \eta(x) dP_\mathcal{X}(x)$. We can define $P_{\mathcal{Y}|x}(B) := \eta(x)$. Note that $P_{\mathcal{Y}|x}(B)$ may not satisfy the σ-additivity in general. For details on conditional probability, see, for example, Shiryaev (1995, §9).

The conditional probability $P_{\mathcal{Y}|x}(B)$ can be uniquely determined only almost surely with respect to P_X. It is thus possible to define Q appropriately only if assumption (A) holds.

In ordinary Bayesian inference, we need only the conditional probability density (likelihood) $p(y|x)$ and prior $\pi(x)$, and not the joint distribution P. In kernel methods, however, the information on the relation between variables is expressed by covariance, which leads to finite sample estimates in terms of Gram matrices, as we see below. It is then necessary to assume the existence of the variable (X,Y) on $\mathcal{X}\times\mathcal{Y}$ with probability P, which gives the conditional probability $P_{\mathcal{Y}|x}$ by conditioning on $X=x$.

Let $k_{\mathcal{X}}$ and $k_{\mathcal{Y}}$ be positive definite kernels on $\mathcal{X}$ and $\mathcal{Y}$, respectively, with respective RKHS $\mathcal{H}_{\mathcal{X}}$ and $\mathcal{H}_{\mathcal{Y}}$. The goal of this subsection is to derive an estimator of the kernel mean of posterior $m_{Q_{X|y}}$. The following theorem is fundamental to discuss conditional probabilities with positive definite kernels.

Theorem 1 (Fukumizu et al., 2004) *If $E[g(Y)|X=\cdot]\in\mathcal{H}_{\mathcal{X}}$ holds[2] for $g\in\mathcal{H}_{\mathcal{Y}}$, then*

$$C_{XX}E[g(Y)|X=\cdot]=C_{XY}g.$$

The above relation motivates to introduce a regularized approximation of the conditional expectation

$$\big(C_{XX}+\varepsilon I\big)^{-1}C_{XY}g,$$

which is shown to converge to $E[g(Y)|X=\cdot]$ in $\mathcal{H}_{\mathcal{X}}$ under appropriate assumptions, as we will see later.

Using Theorem 1, we have the following result, which expresses the kernel mean of $Q_{\mathcal{Y}}$, and implements the Sum Rule in terms of mean embeddings.

Theorem 2 (Song et al., 2009, Equation 6) *Let m_Π and $m_{Q_{\mathcal{Y}}}$ be the kernel means of Π in $\mathcal{H}_{\mathcal{X}}$ and $Q_{\mathcal{Y}}$ in $\mathcal{H}_{\mathcal{Y}}$, respectively. If C_{XX} is injective,[3] $m_\Pi\in\mathcal{R}(C_{XX})$, and $E[g(Y)|X=\cdot]\in\mathcal{H}_{\mathcal{X}}$ for any $g\in\mathcal{H}_{\mathcal{Y}}$, then*

$$m_{Q_{\mathcal{Y}}}=C_{YX}C_{XX}^{-1}m_\Pi, \tag{7}$$

where $C_{XX}^{-1}m_\Pi$ denotes the function mapped to m_Π by C_{XX}.

Proof Take $f\in\mathcal{H}_{\mathcal{X}}$ such that $C_{XX}f=m_\Pi$. It follows from Theorem 1 that for any $g\in\mathcal{H}_{\mathcal{Y}}$, $\langle C_{YX}f,g\rangle=\langle f,C_{XY}g\rangle=\langle f,C_{XX}E[g(Y)|X=\cdot]\rangle=\langle C_{XX}f,E[g(Y)|X=\cdot]\rangle=\langle m_\Pi,E[g(Y)|X=\cdot]\rangle=\langle m_{Q_{\mathcal{Y}}},g\rangle$, which implies $C_{YX}f=m_{Q_{\mathcal{Y}}}$. ■

As discussed by Song et al. (2009), we can regard the operator $C_{YX}C_{XX}^{-1}$ as the kernel expression of the conditional probability $P_{\mathcal{Y}|x}$ or $p(y|x)$. Note, however, that the assumptions $m_\Pi\in\mathcal{R}(C_{XX})$ and $E[g(Y)|X=\cdot]\in\mathcal{H}_{\mathcal{X}}$ may not hold in general; we can easily give counterexamples for the latter in the case of Gaussian kernels.[4] A regularized inverse $(C_{XX}+\varepsilon I)^{-1}$ can be used to remove this

2. The assumption "$E[g(Y)|X=\cdot]\in\mathcal{H}_{\mathcal{X}}$" means that a version of the conditional expectation $E[g(Y)|X=x]$ is included in $\mathcal{H}_{\mathcal{X}}$ as a function of x.
3. Noting $\langle C_{XX}f,f\rangle=E[f(X)^2]$, it is easy to see that C_{XX} is injective if $\mathcal{X}$ is a topological space, $k_{\mathcal{X}}$ is a continuous kernel, and $\mathrm{Supp}(P_X)=\mathcal{X}$, where $\mathrm{Supp}(P_X)$ is the support of P_X.
4. Suppose that $\mathcal{H}_{\mathcal{X}}$ and $\mathcal{H}_{\mathcal{Y}}$ are given by Gaussian kernel, and that X and Y are independent. Then, $E[g(Y)|X=x]$ is a constant function of x, which is known not to be included in a RKHS given by a Gaussian kernel (Steinwart and Christmann, 2008, Corollary 4.44).

strong assumption. An alternative way of obtaining the regularized conditional mean embedding is as the solution to a vector-valued ridge regression problem, as proposed by Grünewälder et al. (2012) and Grünewälder et al. (2013). A connection between conditional embeddings and ridge regression was noted independently by Zhang et al. (2011, Section 3.5). Following Grünewälder et al. (2013, Section 3.2), we seek a bounded linear operator $F : \mathcal{H}_\mathcal{Y} \to \mathcal{H}_\mathcal{X}$ that minimizes the loss

$$\mathcal{E}_c[F] = \sup_{\|h\|_{\mathcal{H}_\mathcal{Y}} \leq 1} E\left(E[h(Y)|X] - \langle Fh, k_\mathcal{X}(X,\cdot)\rangle_{\mathcal{H}_\mathcal{X}}\right)^2,$$

where we take the supremum over the unit ball in $\mathcal{H}_\mathcal{Y}$ to ensure worst-case robustness. Using the Jensen and Cauchy-Schwarz inequalities, we may upper bound this as

$$\mathcal{E}_c[F] \leq E_u[F] := E\|k_\mathcal{Y}(Y,\cdot) - F^*[k_\mathcal{X}(X,\cdot)]\|^2_{\mathcal{H}_\mathcal{Y}},$$

where F^* denotes the adjoint of F. If we stipulate that[5] $F^* \in \mathcal{H}_\mathcal{Y} \otimes \mathcal{H}_\mathcal{X}$, and regularize by the squared Hilbert-Schmidt norm of F^*, we obtain the ridge regression problem

$$\operatorname*{argmin}_{F^* \in \mathcal{H}_\mathcal{Y} \otimes \mathcal{H}_\mathcal{X}} E\|k_\mathcal{Y}(Y,\cdot) - F^*[k_\mathcal{X}(X,\cdot)]\|^2_{\mathcal{H}_\mathcal{Y}} + \varepsilon\|F^*\|^2_{\mathrm{HS}},$$

which has as its solution the regularized kernel conditional mean embedding. In the following, we nonetheless use the unregularized version to derive a population expression of Bayes' rule, use it as a prototype for defining an empirical estimator, and prove its consistency.

Equation (7) has a simple interpretation if the probability density function or Radon-Nikodym derivative $d\Pi/dP_\mathcal{X}$ is included in $\mathcal{H}_\mathcal{X}$ under Assumption (A). From Equation (3) we have $m_\Pi(x) = \int k_\mathcal{X}(x,\tilde{x})d\Pi(\tilde{x}) = \int k_\mathcal{X}(x,\tilde{x})(d\Pi/dP_\mathcal{X})(\tilde{x})dP_\mathcal{X}(\tilde{x})$, which implies $C_{XX}^{-1}m_\Pi = d\Pi/dP_\mathcal{X}$ from Equation (4) under the injective assumption of C_{XX}. Thus Equation (7) is an operator expression of the obvious relation

$$\int\int k_\mathcal{Y}(y,\tilde{y})dP_{\mathcal{Y}|x}(\tilde{y})d\Pi(\tilde{x}) = \int k_\mathcal{Y}(y,\tilde{y})\Big(\frac{d\Pi}{dP_\mathcal{X}}\Big)(\tilde{x})dP(\tilde{x},\tilde{y}).$$

In many applications of Bayesian inference, the probability conditioned on a particular value should be computed. By plugging the point measure at x into Π in Equation (7), we have a population expression[6]

$$E[k_\mathcal{Y}(\cdot,Y)|X = x] = C_{YX}{C_{XX}}^{-1}k_\mathcal{X}(\cdot,x), \tag{8}$$

or more rigorously we can consider a regularized inversion

$$E^{reg}_\varepsilon[k_\mathcal{Y}(\cdot,Y)|X = x] := C_{YX}(C_{XX} + \varepsilon I)^{-1}k_\mathcal{X}(\cdot,x) \tag{9}$$

as an approximation of the conditional expectation $E[k_\mathcal{Y}(\cdot,Y)|X = x]$. Note that in the latter expression we do not need to assume $k_\mathcal{X}(\cdot,x) \in \mathcal{R}(C_{XX})$, which is too strong in many situations. We

5. Note that more complex vector-valued RKHS are possible; see, for example, Micchelli and Pontil (2005).

6. The expression Equation (8) has been considered in Song et al. (2009, 2010a) as the kernel mean of the conditional probability. It must be noted that for this case the assumption $m_\Pi = k(\cdot,x) \in \mathcal{R}(C_{XX})$ in Theorem 2 may not hold in general. Suppose $C_{XX}h_x = k_\mathcal{X}(\cdot,x)$ were to hold for some $h_x \in \mathcal{H}_\mathcal{X}$. Taking the inner product with $k_\mathcal{X}(\cdot,\tilde{x})$ would then imply $k_\mathcal{X}(x,\tilde{x}) = \int h_x(x')k_\mathcal{X}(\tilde{x},x')dP_\mathcal{X}(x')$, which is not possible for many popular kernels, including the Gaussian kernel.

will show in Theorem 8 that under some mild conditions a regularized empirical estimator based on Equation (9) is a consistent estimator of $E[k_{\mathcal{Y}}(\cdot,Y)|X=x]$.

To derive kernel realization of Bayes' rule, suppose that we know the covariance operators C_{ZW} and C_{WW} for the random variable $(Z,W) \sim Q$, where Q is defined by Equation (5). The conditional probability $E[k_{\mathcal{X}}(\cdot,Z)|W=y]$ is then exactly the kernel mean of the posterior distribution for observation $y \in \mathcal{Y}$. Equation (9) gives the regularized approximate of the kernel mean of posterior;

$$C_{ZW}(C_{WW}+\delta I)^{-1}k_{\mathcal{Y}}(\cdot,y), \tag{10}$$

where δ is a positive regularization constant. The remaining task is thus to derive the covariance operators C_{ZW} and C_{WW}. This can be done by recalling that the kernel mean $m_Q = m_{(ZW)} \in \mathcal{H}_{\mathcal{X}} \otimes \mathcal{H}_{\mathcal{Y}}$ can be identified with the covariance operator $C_{ZW} : \mathcal{H}_{\mathcal{Y}} \to \mathcal{H}_{\mathcal{X}}$ by the standard identification of a tensor $\sum_i f_i \otimes g_i$ ($f_i \in \mathcal{H}_{\mathcal{X}}$ and $g_i \in \mathcal{H}_{\mathcal{Y}}$) and a Hilbert-Schmidt operator $h \mapsto \sum_i g_i \langle f_i, h\rangle_{\mathcal{H}_{\mathcal{X}}}$. From Theorem 2, the kernel mean m_Q is given by the following tensor representation:

$$m_Q = C_{(YX)X}C_{XX}^{-1}m_\Pi \in \mathcal{H}_{\mathcal{Y}} \otimes \mathcal{H}_{\mathcal{X}},$$

where the covariance operator $C_{(YX)X} : \mathcal{H}_{\mathcal{X}} \to \mathcal{H}_{\mathcal{Y}} \otimes \mathcal{H}_{\mathcal{X}}$ is defined by the random variable $((Y,X),X)$ taking values on $(\mathcal{Y}\times\mathcal{X})\times\mathcal{X}$. We can alternatively and more generally use an approximation by the regularized inversion $m_Q^{reg} = C_{(YX)X}(C_{XX}+\delta)^{-1}m_\Pi$, as in Equation (9). This expression provides the covariance operator C_{ZW}. Similarly, the kernel mean $m_{(WW)}$ on the product space $\mathcal{H}_{\mathcal{Y}} \otimes \mathcal{H}_{\mathcal{Y}}$ is identified with C_{WW}, and the expression

$$m_{(WW)} = C_{(YY)X}C_{XX}^{-1}m_\Pi$$

gives a way of estimating the operator C_{WW}.

The above argument can be rigorously implemented, if empirical estimators are considered. Let $(X_1,Y_1),\ldots,(X_n,Y_n)$ be an i.i.d. sample with law P. Since we need to express the information in the variables in terms of Gram matrices given by data points, we assume the prior is also expressed in the form of an empirical estimate, and that we have a consistent estimator of m_Π in the form

$$\widehat{m}_\Pi^{(\ell)} = \sum_{j=1}^{\ell} \gamma_j k_{\mathcal{X}}(\cdot,U_j),$$

where $U_1,\ldots,U_\ell$ are points in $\mathcal{X}$ and γ_j are the weights. The data points U_j may or may not be a sample from the prior Π, and negative values are allowed for γ_j. Such negative weights may appear in successive applications of the kernel Bayes rule, as in the state-space example of Section 4.3. Based on Equation (9), the empirical estimators for $m_{(ZW)}$ and $m_{(WW)}$ are defined respectively by

$$\widehat{m}_{(ZW)} = \widehat{C}_{(YX)X}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\widehat{m}_\Pi^{(\ell)}, \quad \widehat{m}_{(WW)} = \widehat{C}_{(YY)X}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\widehat{m}_\Pi^{(\ell)},$$

where ε_n is the coefficient of the Tikhonov-type regularization for operator inversion, and I is the identity operator. The empirical estimators $\widehat{C}_{ZW}$ and $\widehat{C}_{WW}$ for C_{ZW} and C_{WW} are identified with $\widehat{m}_{(ZW)}$ and $\widehat{m}_{(WW)}$, respectively. In the following, G_X and G_Y denote the Gram matrices $(k_{\mathcal{X}}(X_i,X_j))$ and $(k_{\mathcal{Y}}(Y_i,Y_j))$, respectively, and I_n is the identity matrix of size n.

Proposition 3 *The Gram matrix expressions of $\widehat{C}_{ZW}$ and $\widehat{C}_{WW}$ are given by*

$$\widehat{C}_{ZW} = \sum_{i=1}^{n} \widehat{\mu}_i k_{\mathcal{X}}(\cdot, X_i) \otimes k_{\mathcal{Y}}(\cdot, Y_i) \quad \textit{and} \quad \widehat{C}_{WW} = \sum_{i=1}^{n} \widehat{\mu}_i k_{\mathcal{Y}}(\cdot, Y_i) \otimes k_{\mathcal{Y}}(\cdot, Y_i),$$

respectively, where the common coefficient $\widehat{\mu} \in \mathbb{R}^n$ is

$$\widehat{\mu} = \Big(\frac{1}{n} G_X + \varepsilon_n I_n\Big)^{-1} \widehat{\mathbf{m}}_{\Pi}, \quad \widehat{\mathbf{m}}_{\Pi,i} = \widehat{m}_{\Pi}(X_i) = \sum_{j=1}^{\ell} \gamma_j k_{\mathcal{X}}(X_i, U_j). \tag{11}$$

The proof is similar to that of Proposition 4 below, and is omitted. The expressions in Proposition 3 imply that the probabilities Q and $Q_{\mathcal{Y}}$ are estimated by the weighted samples $\{((X_i, Y_i), \widehat{\mu}_i)\}_{i=1}^n$ and $\{(Y_i, \widehat{\mu}_i)\}_{i=1}^n$, respectively, with common weights. Since the weight $\widehat{\mu}_i$ may be negative, the operator inversion $(\widehat{C}_{WW} + \delta_n I)^{-1}$ in Equation (10) may be impossible or unstable. We thus use another type of Tikhonov regularization,[7] resulting in the estimator

$$\widehat{m}_{Q_{X|y}} := \widehat{C}_{ZW} \big(\widehat{C}_{WW}^2 + \delta_n I\big)^{-1} \widehat{C}_{WW} k_{\mathcal{Y}}(\cdot, y). \tag{12}$$

Proposition 4 *For any $y \in \mathcal{Y}$, the Gram matrix expression of $\widehat{m}_{Q_{X|y}}$ is given by*

$$\widehat{m}_{Q_{X|y}} = \mathbf{k}_X^T R_{X|Y} \mathbf{k}_Y(y), \qquad R_{X|Y} := \Lambda G_Y \big((\Lambda G_Y)^2 + \delta_n I_n\big)^{-1} \Lambda, \tag{13}$$

where $\Lambda = \mathrm{diag}(\widehat{\mu})$ is a diagonal matrix with elements $\widehat{\mu}_i$ in Equation (11), and $\mathbf{k}_X \in {\mathcal{H}_{\mathcal{X}}}^n := \mathcal{H}_{\mathcal{X}} \times \cdots \times \mathcal{H}_{\mathcal{X}}$ (n direct product) and $\mathbf{k}_Y \in {\mathcal{H}_{\mathcal{Y}}}^n := \mathcal{H}_{\mathcal{Y}} \times \cdots \times \mathcal{H}_{\mathcal{Y}}$ are given by

$$\mathbf{k}_X = (k_{\mathcal{X}}(\cdot, X_1), \ldots, k_{\mathcal{X}}(\cdot, X_n))^T \quad \textit{and} \quad \mathbf{k}_Y = (k_{\mathcal{Y}}(\cdot, Y_1), \ldots, k_{\mathcal{Y}}(\cdot, Y_n))^T.$$

Proof Let $h = (\widehat{C}_{WW}^2 + \delta_n I)^{-1} \widehat{C}_{WW} k_{\mathcal{Y}}(\cdot, y)$, and decompose it as $h = \sum_{i=1}^n \alpha_i k_{\mathcal{Y}}(\cdot, Y_i) + h_\perp = \alpha^T \mathbf{k}_Y + h_\perp$, where $h_\perp$ is orthogonal to $\mathrm{Span}\{k_{\mathcal{Y}}(\cdot, Y_i)\}_{i=1}^n$. Expansion of $(\widehat{C}_{WW}^2 + \delta_n I)h = \widehat{C}_{WW} k_{\mathcal{Y}}(\cdot, y)$ gives $\mathbf{k}_Y^T (\Lambda G_Y)^2 \alpha + \delta_n \mathbf{k}_Y^T \alpha + \delta_n h_\perp = \mathbf{k}_Y^T \Lambda \mathbf{k}_Y(y)$. Taking the inner product with $k_{\mathcal{Y}}(\cdot, Y_j)$, we have

$$\big((G_Y \Lambda)^2 + \delta_n I_n\big) G_Y \alpha = G_Y \Lambda \mathbf{k}_Y(y).$$

The coefficient ρ in $\widehat{m}_{Q_{X|y}} = \widehat{C}_{ZW} h = \sum_{i=1}^n \rho_i k_{\mathcal{X}}(\cdot, X_i)$ is given by $\rho = \Lambda G_Y \alpha$, and thus

$$\rho = \Lambda \big((G_Y \Lambda)^2 + \delta_n I_n\big)^{-1} G_Y \Lambda \mathbf{k}_Y(y) = \Lambda G_Y \big((\Lambda G_Y)^2 + \delta_n I_n\big)^{-1} \Lambda \mathbf{k}_Y(y).$$

■

We call Equations (12) and (13) the *kernel Bayes' rule* (KBR). The required computations are summarized in Figure 1. The KBR uses a weighted sample to represent the posterior; it is similar in this respect to sampling methods such as importance sampling and sequential Monte Carlo (Doucet et al., 2001). The KBR method, however, does not generate samples of the posterior, but updates the weights of a sample by matrix computation. Note also that the weights in KBR may take negative values. The interpretation as a probability is then not straightforward, hence the mean

7. An alternative thresholding approach is proposed by Nishiyama et al. (2012), although its consistency remains to be established.

Input: (i) $\{(X_i, Y_i)\}_{i=1}^n$: sample to express P. (ii) $\{(U_j, \gamma_j)\}_{j=1}^\ell$: weighted sample to express the kernel mean of the prior $\widehat{m}_\Pi$. (iii) ε_n, δ_n: regularization constants.

Computation:

1. Compute Gram matrices $G_X = (k_\mathcal{X}(X_i, X_j))$, $G_Y = (k_\mathcal{Y}(Y_i, Y_j))$, and a vector $\widehat{\mathbf{m}}_\Pi = (\sum_{j=1}^\ell \gamma_j k_\mathcal{X}(X_i, U_j))_{i=1}^n$.
2. Compute $\widehat{\mu} = n(G_X + n\varepsilon_n I_n)^{-1} \widehat{\mathbf{m}}_\Pi$.
   [If the inversion fails, increase ε_n by $\varepsilon_n := c\varepsilon_n$ with $c > 1$.]
3. Compute $R_{X|Y} = \Lambda G_Y ((\Lambda G_Y)^2 + \delta_n I_n)^{-1} \Lambda$, where $\Lambda = \mathrm{diag}(\widehat{\mu})$.
   [If the inversion fails, increase δ_n by $\delta_n := c\delta_n$ with $c > 1$.]

Output: $n \times n$ matrix $R_{X|Y}$.
Given conditioning value y, the kernel mean of the posterior $q(x|y)$ is estimated by the weighted sample $\{(X_i, \rho_i)\}_{i=1}^n$ with weights $\rho = R_{X|Y} \mathbf{k}_Y(y)$, where $\mathbf{k}_Y(y) = (k_\mathcal{Y}(Y_i, y))_{i=1}^n$.

Figure 1: Algorithm for Kernel Bayes' Rule

embedding viewpoint should take precedence (i.e., even if some weights are negative, we may still use result of KBR to estimate posterior expectations of RKHS functions). We will give experimental comparisons between KBR and sampling methods in Section 5.1.

If our aim is to estimate the expectation of a function $f \in \mathcal{H}_\mathcal{X}$ with respect to the posterior, the reproducing property of Equation (2) gives an estimator

$$\langle f, \widehat{m}_{Q_{X|y}} \rangle_{\mathcal{H}_\mathcal{X}} = \mathbf{f}_X^T R_{X|Y} \mathbf{k}_\mathcal{Y}(y), \tag{14}$$

where $\mathbf{f}_X = (f(X_1), \ldots, f(X_n))^T \in \mathbb{R}^n$.

3.2 Consistency of the KBR Estimator

We now demonstrate the consistency of the KBR estimator in Equation (14). We first show consistency of the estimator, and next the rate of consistency under stronger conditions.

Theorem 5 *Let (X,Y) be a random variable on $\mathcal{X} \times \mathcal{Y}$ with distribution P, (Z,W) be a random variable on $\mathcal{X} \times \mathcal{Y}$ such that the distribution is Q defined by Equation (5), and $\widehat{m}_\Pi^{(\ell_n)}$ be a consistent estimator of m_Π in $\mathcal{H}_\mathcal{X}$ norm. Assume that C_{XX} is injective and that $E[k_\mathcal{Y}(Y,\tilde{Y})|X = x, \tilde{X} = \tilde{x}]$ and $E[k_\mathcal{X}(Z,\tilde{Z})|W = y, \tilde{W} = \tilde{y}]$ are included in the product spaces $\mathcal{H}_\mathcal{X} \otimes \mathcal{H}_\mathcal{X}$ and $\mathcal{H}_\mathcal{Y} \otimes \mathcal{H}_\mathcal{Y}$, respectively, as a function of $(x,\tilde{x})$ and $(y,\tilde{y})$, where $(\tilde{X},\tilde{Y})$ and $(\tilde{Z},\tilde{W})$ are independent copies of (X,Y) and (Z,W), respectively. Then, for any sufficiently slow decay of the regularization coefficients ε_n and δ_n, we have for any $y \in \mathcal{Y}$*

$$\left\| \mathbf{k}_X^T R_{X|Y} \mathbf{k}_Y(y) - m_{Q_{X|y}} \right\|_{\mathcal{H}_\mathcal{X}} \to 0$$

in probability as $n \to \infty$, where $\mathbf{k}_X^T R_{X|Y} \mathbf{k}_Y(y)$ is the KBR estimator given by Equation (13) and $m_{Q_{X|y}} = E[k_\mathcal{X}(\cdot, Z)|W = y]$ is the kernel mean of posterior given $W = y$.

It is obvious from the reproducing property that this theorem also guarantees the consistency of the posterior expectation in Equation (14). The rate of decrease of ε_n and δ_n depends on the convergence rate of $\widehat{m}_\Pi^{(\ell_n)}$ and other smoothness assumptions.

Next, we show convergence rates of the KBR estimator for the expectation with posterior under stronger assumptions. In the following two theorems, we show only the rates that can be derived under certain specific assumptions, and defer more detailed discussions and proofs to Section 6. We assume here that the sample size $\ell = \ell_n$ for the prior goes to infinity as the sample size n for the likelihood goes to infinity, and that $\widehat{m}_\Pi^{(\ell_n)}$ is n^α-consistent in RKHS norm.

Theorem 6 *Let f be a function in $\mathcal{H}_\mathcal{X}$, (X,Y) be a random variable on $\mathcal{X}\times\mathcal{Y}$ with distribution P, (Z,W) be a random variable on $\mathcal{X}\times\mathcal{Y}$ with the distribution Q defined by Equation (5), and $\widehat{m}_\Pi^{(\ell_n)}$ be an estimator of m_Π such that $\|\widehat{m}_\Pi^{(\ell_n)} - m_\Pi\|_{\mathcal{H}_\mathcal{X}} = O_p(n^{-\alpha})$ as $n\to\infty$ for some $0<\alpha\le 1/2$. Assume that the Radon Nikodym derivative $d\Pi/dP_\mathcal{X}$ is included in $\mathcal{R}(C_{XX}^{1/2})$, and $E[f(Z)|W=\cdot]\in\mathcal{R}(C_{WW}^2)$. With the regularization constants $\varepsilon_n = n^{-\frac{2}{3}\alpha}$ and $\delta_n = n^{-\frac{8}{27}\alpha}$, we have for any $y\in\mathcal{Y}$*

$$\mathbf{f}_X^T R_{X|Y}\mathbf{k}_Y(y) - E[f(Z)|W=y] = O_p(n^{-\frac{8}{27}\alpha}), \quad (n\to\infty),$$

where $\mathbf{f}_X^T R_{X|Y}\mathbf{k}_Y(y)$ is given by Equation (14).

It is possible to extend the covariance operator C_{WW} to one defined on $L^2(Q_\mathcal{Y})$ by

$$\tilde{C}_{WW}\phi = \int k_\mathcal{Y}(y,w)\phi(w)dQ_\mathcal{Y}(w), \qquad (\phi\in L^2(Q_\mathcal{Y})). \tag{15}$$

If we consider the convergence on average over y, we have a slightly better rate on the consistency of the KBR estimator in $L^2(Q_\mathcal{Y})$.

Theorem 7 *Let f be a function in $\mathcal{H}_\mathcal{X}$, (Z,W) be a random vector on $\mathcal{X}\times\mathcal{Y}$ with the distribution Q defined by Equation (5), and $\widehat{m}_\Pi^{(\ell_n)}$ be an estimator of m_Π such that $\|\widehat{m}_\Pi^{(\ell_n)} - m_\Pi\|_{\mathcal{H}_\mathcal{X}} = O_p(n^{-\alpha})$ as $n\to\infty$ for some $0<\alpha\le 1/2$. Assume that the Radon Nikodym derivative $d\Pi/dP_\mathcal{X}$ is included in $\mathcal{R}(C_{XX}^{1/2})$, and $E[f(Z)|W=\cdot]\in\mathcal{R}(\tilde{C}_{WW}^2)$. With the regularization constants $\varepsilon_n = n^{-\frac{2}{3}\alpha}$ and $\delta_n = n^{-\frac{1}{3}\alpha}$, we have*

$$\left\|\mathbf{f}_X^T R_{X|Y}\mathbf{k}_Y(W) - E[f(Z)|W]\right\|_{L^2(Q_\mathcal{Y})} = O_p(n^{-\frac{1}{3}\alpha}), \quad (n\to\infty).$$

The condition $d\Pi/dP_\mathcal{X}\in\mathcal{R}(C_{XX}^{1/2})$ requires the prior to be sufficiently smooth. If $\widehat{m}_\Pi^{(\ell_n)}$ is a direct empirical mean with an i.i.d. sample of size n from Π, typically $\alpha = 1/2$, with which the theorems imply $n^{4/27}$-consistency for every y, and $n^{1/6}$-consistency in the $L^2(Q_\mathcal{Y})$ sense. While these might seem to be slow rates, the rate of convergence can in practice be much faster than the above theoretical guarantees.

While the convergence rates shown in the above theorems do not depend on the dimensionality of original spaces, the rates may not be optimal. In fact, in the case of kernel ridge regression, the optimal rates are known under additional information on the spectrum of covariance operators (Caponnetto and De Vito, 2007). It is also known (Eberts and Steinwart, 2011) that, given the target function is in the Sobolev space of order α, the convergence rates is arbitrary close to $O_p(n^{-2\alpha/(2\alpha+d)})$, the best rate for any linear estimator (Stone, 1982), where d is the dimensionality of the predictor. Similar convergence rates for KBR incorporating the information on eigenspectrum or smoothness will be interesting future works, in the light of the equivalence of the conditional mean embedding and operator-valued regression shown by Grünewälder et al. (2012) and Grünewälder et al. (2013).

4. Bayesian Inference with Kernel Bayes' Rule

We discuss problem settings for which KBR may be applied in Section 4.1. We then provide notes on practical implementation in Section 4.2, including a cross-validation procedure for parameter selection and suggestions for speeding computation. In Section 4.3, we apply KBR to the filtering problem in a nonparametric state-space model. Finally, in Section 4.4, we give a brief overview of Approximate Bayesian Computation (ABC), a widely used sample-based method which applies to similar problem domains.

4.1 Applications of Kernel Bayes' Rule

In Bayesian inference, we are usually interested in finding a point estimate such as the MAP solution, the expectation of a function under the posterior, or other properties of the distribution. Given that KBR provides a posterior estimate in the form of a kernel mean (which uniquely determines the distribution when a characteristic kernel is used), we now describe how our kernel approach applies to problems in Bayesian inference.

First, we have already seen that under appropriate assumptions, a consistent estimator for the expectation of $f \in \mathcal{H}_X$ can be defined with respect to the posterior. On the other hand, unless $f \in \mathcal{H}_X$ holds, there is no theoretical guarantee that it gives a good estimate. In Section 5.1, we discuss experimental results observed in these situations.

To obtain a point estimate of the posterior on x, Song et al. (2009) propose to use the preimage $\widehat{x} = \arg\min_x \|k_X(\cdot,x) - \mathbf{k}_X^T R_{X|Y} \mathbf{k}_Y(y)\|^2_{\mathcal{H}_X}$, which represents the posterior mean most effectively by one point. We use this approach in the present paper when point estimates are sought. In the case of the Gaussian kernel $\exp(-\|x-y\|^2/(2\sigma^2))$, the fixed point method

$$x^{(t+1)} = \frac{\sum_{i=1}^n X_i \rho_i \exp(-\|X_i - x^{(t)}\|^2/(2\sigma^2))}{\sum_{i=1}^n \rho_i \exp(-\|X_i - x^{(t)}|^2/(2\sigma^2))},$$

where $\rho = R_{X|Y}\mathbf{k}_Y(y)$, can be used to optimize x sequentially (Mika et al., 1999). This method usually converges very fast, although no theoretical guarantee exists for the convergence to the globally optimal point, as is usual in non-convex optimization.

A notable property of KBR is that the prior and likelihood are represented in terms of samples. Thus, unlike many approaches to Bayesian inference, precise knowledge of the prior and likelihood distributions is not needed, once samples are obtained. The following are typical situations where the KBR approach is advantageous:

- The probabilistic relation among variables is difficult to realize with a simple parametric model, while we can obtain samples of the variables easily. We will see such an example in Section 4.3.

- The probability density function of the prior and/or likelihood is hard to obtain explicitly, but sampling is possible:

  - In the field of population genetics, Bayesian inference is used with a likelihood expressed by branching processes to model the split of species, for which the explicit density is hard to obtain. Approximate Bayesian Computation (ABC) is a popular method for approximately sampling from a posterior without knowing the functional form (Tavaré et al., 1997; Marjoram et al., 2003; Sisson et al., 2007).

– In nonparametric Bayesian inference (Müller and Quintana, 2004 and references therein), the prior is typically given in the form of a process without a density form. In this case, sampling methods are often applied (MacEachern, 1994; West et al., 1994; MacEachern et al., 1999, among others). Alternatively, the posterior may be approximated using variational methods (Blei and Jordan, 2006).

We will present an experimental comparison of KBR and ABC in Section 5.2.

- Even if explicit forms for the likelihood and prior are available, and standard sampling methods such as MCMC or sequential MC are applicable, the computation of a posterior estimate given y might still be computationally costly, making real-time applications infeasible. Using KBR, however, the expectation of a function of the posterior given different y is obtained simply by taking the inner product as in Equation (14), once $\mathbf{f}_X^T R_{X|Y}$ has been computed.

4.2 Discussions Concerning Implementation

When implementing KBR, a number of factors should be borne in mind to ensure good performance. First, in common with many nonparametric approaches, KBR requires training data in the region of the new "test" points for results to be meaningful. In other words, if the point on which we condition appears in a region far from the sample used for the estimation, the posterior estimator will be unreliable.

Second, when computing the posterior in KBR, Gram matrix inversion is necessary, which would cost $O(n^3)$ for sample size n if attempted directly. Substantial cost reductions can be achieved if the Gram matrices are replaced by low rank matrix approximations. A popular choice is the incomplete Cholesky factorization (Fine and Scheinberg, 2001), which approximates a Gram matrix in the form of $\Gamma\Gamma^T$ with $n \times r$ matrix Γ ($r \ll n$) at cost $O(nr^2)$. Using this and the Woodbury identity, the KBR can be approximately computed at cost $O(nr^2)$, which is linear in the sample size n. It is known that in some typical cases the eigenspectram of a Gram matrix decays fast (Widom, 1963, 1964; Bach and Jordan, 2002). We can therefore expect that the incomplete Cholesky factorization to reduce the computational cost effectively without much degrading estimation accuracy.

Third, kernel choice or model selection is key to effective performance of any kernel method. In the case of KBR, we have three model parameters: the kernel (or its parameter, e.g., the bandwidth), and the regularization parameters ε_n, and δ_n. The strategy for parameter selection depends on how the posterior is to be used in the inference problem. If it is to be applied in regression or classification, we can use standard cross-validation. In the filtering experiments in Section 5, we use a validation method where we divide the training sample in two.

A more general model selection approach can also be formulated, by creating a new regression problem for the purpose. Suppose the prior Π is given by the marginal P_X of P. The posterior $Q_{X|y}$ averaged with respect to P_Y is then equal to the marginal P_X itself. We are thus able to compare the discrepancy of the empirical kernel mean of P_X and the average of the estimators $\widehat{m}_{Q_{X|y=Y_i}}$ over Y_i. This leads to a K-fold cross validation approach: for a partition of $\{1,\ldots,n\}$ into K disjoint subsets $\{T_a\}_{a=1}^K$, let $\widehat{m}_{Q_{X|y}}^{[-a]}$ be the kernel mean of posterior computed using Gram matrices on data $\{(X_i,Y_i)\}_{i\notin T_a}$, and based on the prior mean $\widehat{m}_X^{[-a]}$ with data $\{X_i\}_{i\notin T_a}$. We can then cross validate by minimizing $\sum_{a=1}^K \left\| \frac{1}{|T_a|}\sum_{j\in T_a} \widehat{m}_{Q_{X|y=Y_j}}^{[-a]} - \widehat{m}_X^{[a]} \right\|_{\mathcal{H}_X}^2$, where $\widehat{m}_X^{[a]} = \frac{1}{|T_a|}\sum_{j\in T_a} k_X(\cdot, X_j)$.

4.3 Application to a Nonparametric State-Space Model

We next describe how KBR may be used in a particular application: namely, inference in a general time invariant state-space model,

$$p(X,Y) = \pi(X_1)\prod_{t=1}^{T+1} p(Y_t|X_t)\prod_{t=1}^{T} q(X_{t+1}|X_t),$$

where Y_t is an observable variable, and X_t is a hidden state variable. We begin with a brief review of alternative strategies for inference in state-space models with complex dynamics, for which linear models are not suitable. The extended Kalman filter (EKF) and unscented Kalman filter (UKF, Julier and Uhlmann, 1997) are nonlinear extensions of the standard linear Kalman filter, and are well established in this setting. Alternatively, nonparametric estimates of conditional density functions can be employed, including kernel density estimation or distribution estimates on a partitioning of the space (Monbet et al., 2008; Thrun et al., 1999). Kernel density estimates converges more slowly in L^2 as the dimension d of the space increases, however: for an optimal bandwidth choice, this error drops as $O(n^{-4(4+d)})$ (Wasserman, 2006, Section 6.5). If the goal is to take expectations of smooth functions (i.e., RKHS functions), rather than to obtain consistent density estimates, then we can expect better performance. We show our posterior estimate of the expectation of an RKHS function converges at a rate independent of d. Most relevant to this paper are Song et al. (2009) and Song et al. (2010b), in which the kernel means and covariance operators are used to implement a nonparametric HMM.

In this paper, we apply the KBR for inference in the nonparametric state-space model. We do not assume the conditional probabilities $p(Y_t|X_t)$ and $q(X_{t+1}|X_t)$ to be known explicitly, nor do we estimate them with simple parametric models. Rather, we assume a sample $(X_1,Y_1,\ldots,X_{T+1},Y_{T+1})$ is given for both the observable and hidden variables in the training phase. The conditional probability for observation process $p(y|x)$ and the transition $q(x_{t+1}|x_t)$ are represented by the empirical covariance operators as computed on the training sample,

$$\widehat{C}_{XY} = \frac{1}{T}\sum_{i=1}^{T} k_{\mathcal{X}}(\cdot,X_i)\otimes k_{\mathcal{Y}}(\cdot,Y_i), \quad \widehat{C}_{X_{+1}X} = \frac{1}{T}\sum_{i=1}^{T} k_{\mathcal{X}}(\cdot,X_{i+1})\otimes k_{\mathcal{X}}(\cdot,X_i), \tag{16}$$

$$\widehat{C}_{YY} = \frac{1}{T}\sum_{i=1}^{T} k_{\mathcal{Y}}(\cdot,Y_i)\otimes k_{\mathcal{Y}}(\cdot,Y_i), \quad \widehat{C}_{XX} = \frac{1}{T}\sum_{i=1}^{T} k_{\mathcal{X}}(\cdot,X_i)\otimes k_{\mathcal{X}}(\cdot,X_i).$$

While the sample is not i.i.d., it is known that the empirical covariances converge to the covariances with respect to the stationary distribution as $T\to\infty$, under a mixing condition.[8] We therefore use the above estimator for the covariance operators.

Typical applications of the state-space model are filtering, prediction, and smoothing, which are defined by the estimation of $p(x_s|y_1,\ldots,y_t)$ for $s=t$, $s>t$, and $s<t$, respectively. Using the KBR, any of these can be computed. For simplicity we explain the filtering problem in this paper, but the remaining cases are similar. In filtering, given new observations $\tilde{y}_1,\ldots,\tilde{y}_t$, we wish to estimate the current hidden state x_t. The sequential estimate for the kernel mean of $p(x_t|\tilde{y}_1,\ldots,\tilde{y}_t)$ can be derived via KBR. Suppose we already have an estimator of the kernel mean of $p(x_t|\tilde{y}_1,\ldots,\tilde{y}_t)$ in the form

$$\widehat{m}_{x_t|\tilde{y}_1,\ldots,\tilde{y}_t} = \sum_{s=1}^{T} \alpha_s^{(t)} k_{\mathcal{X}}(\cdot,X_s),$$

8. One such condition to guarantee the central limit theorem for Markov chains in separable Hilbert spaces is *geometrical ergodicity*. See, for example, Merlevéde et al. (1997) and Stachurski (2012) for details.

where $\alpha_i^{(t)} = \alpha_i^{(t)}(\tilde{y}_1, \ldots, \tilde{y}_t)$ are the coefficients at time t. We wish to derive an update rule to obtain $\alpha^{(t+1)}(\tilde{y}_1, \ldots, \tilde{y}_{t+1})$.

For the forward propagation $p(x_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_t) = \int q(x_{t+1}|x_t)p(x_t|\tilde{y}_1, \ldots, \tilde{y}_t)dx_t$, based on Equation (9) the kernel mean of x_{t+1} given $\tilde{y}_1, \ldots, \tilde{y}_t$ is estimated by

$$\widehat{m}_{x_{t+1}|\tilde{y}_1,\ldots,\tilde{y}_t} = \widehat{C}_{X_{+1}X}(\widehat{C}_{XX} + \varepsilon_T I)^{-1}\widehat{m}_{x_t|\tilde{y}_1,\ldots,\tilde{y}_t} = \mathbf{k}_{X_{+1}}^T (G_X + T\varepsilon_T I_T)^{-1} G_X \alpha^{(t)},$$

where $\mathbf{k}_{X_{+1}}^T = (k_{\mathcal{X}}(\cdot, X_2), \ldots, k_{\mathcal{X}}(\cdot, X_{T+1}))$. Using the similar estimator with $p(y_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_t) = \int p(y_{t+1}|x_{t+1})p(x_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_t)dx_t$, we have an estimate for the kernel mean of the prediction $p(y_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_t)$,

$$\widehat{m}_{y_{t+1}|\tilde{y}_1,\ldots,\tilde{y}_t} = \widehat{C}_{YX}(\widehat{C}_{XX} + \varepsilon_T I)^{-1}\widehat{m}_{x_{t+1}|\tilde{y}_1,\ldots,\tilde{y}_t} = \sum_{i=1}^{T} \widehat{\mu}_i^{(t+1)} k_{\mathcal{Y}}(\cdot, Y_i),$$

where the coefficients $\widehat{\mu}^{(t+1)} = (\widehat{\mu}_i^{(t+1)})_{i=1}^T$ are given by

$$\widehat{\mu}^{(t+1)} = \left(G_X + T\varepsilon_T I_T\right)^{-1} G_{XX_{+1}} \left(G_X + T\varepsilon_T I_T\right)^{-1} G_X \alpha^{(t)}. \tag{17}$$

Here $G_{XX_{+1}}$ is the "transfer" matrix defined by $\left(G_{XX_{+1}}\right)_{ij} = k_{\mathcal{X}}(X_i, X_{j+1})$. From

$$p(x_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_{t+1}) = \frac{p(y_{t+1}|x_{t+1})p(x_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_t)}{\int q(y_{t+1}|x_{t+1})p(x_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_t)dx_{t+1}},$$

the kernel Bayes' rule with the prior $p(x_{t+1}|\tilde{y}_1, \ldots, \tilde{y}_t)$ and the likelihood $q(y_{t+1}|x_{t+1})$ yields

$$\alpha^{(t+1)} = \Lambda^{(t+1)} G_Y \left((\Lambda^{(t+1)} G_Y)^2 + \delta_T I_T\right)^{-1} \Lambda^{(t+1)} \mathbf{k}_Y(\tilde{y}_{t+1}), \tag{18}$$

where $\Lambda^{(t+1)} = \mathrm{diag}(\widehat{\mu}_1^{(t+1)}, \ldots, \widehat{\mu}_T^{(t+1)})$. Equations (17) and (18) describe the update rule of $\alpha^{(t)}(\tilde{y}_1, \ldots, \tilde{y}_t)$.

If the prior $\pi(x_1)$ is available, the posterior estimate at x_1 given $\tilde{y}_1$ is obtained by the kernel Bayes' rule. If not, we may use Equation (8) to get an initial estimate $\widehat{C}_{XY}(\widehat{C}_{YY} + \varepsilon_n I)^{-1} k_{\mathcal{Y}}(\cdot, \tilde{y}_1)$, yielding $\alpha^{(1)}(\tilde{y}_1) = T(G_Y + T\varepsilon_T I_T)^{-1}\mathbf{k}_Y(\tilde{y}_1)$.

In sequential filtering, a substantial reduction in computational cost can be achieved by low rank matrix approximations, as discussed in Section 4.2. Given an approximation of rank r for the Gram matrices and transfer matrix, and employing the Woodbury identity, the computation costs just $O(Tr^2)$ for each time step.

4.4 Bayesian Computation Without Likelihood

We next address the setting where the likelihood is not known in analytic form, but sampling is possible. In this case, Approximate Bayesian Computation (ABC) is a popular method for Bayesian inference. The simplest form of ABC, which is called the rejection method, generates an approximate sample from $Q(Z|W = y)$ as follows: (i) generate a sample X_t from the prior Π, (ii) generate a sample Y_t from $P(Y|X_t)$, (iii) if $D(y, Y_t) < \tau$, accept X_t; otherwise reject, (iv) go to (i). In step (iii), D is a distance measure of the space $\mathcal{X}$, and τ is tolerance to acceptance.

In the same setting as ABC, KBR gives the following sampling-based method for computing the kernel posterior mean:

1. Generate a sample $X_1, \ldots, X_n$ from the prior Π.
2. Generate a sample Y_t from $P(Y|X_t)$ $(t = 1, \ldots, n)$.
3. Compute Gram matrices G_X and G_Y with $(X_1, Y_1), \ldots, (X_n, Y_n)$, and $R_{X|Y}\mathbf{k}_Y(y)$.

Alternatively, since (X_t, Y_t) is an sample from Q, it is possible to use simply Equation (8) for the kernel mean of the conditional probability $q(x|y)$. As in Song et al. (2009), the estimator is given by

$$\sum_{t=1}^{n} \nu_j k_X(\cdot, X_t), \quad \nu = (G_Y + n\varepsilon_n I_n)^{-1}\mathbf{k}_Y(y). \tag{19}$$

The distribution of a sample generated by ABC approaches to the true posterior if τ goes to zero, while empirical estimates via the kernel approaches converge to the true posterior mean embedding in the limit of infinite sample size. The efficiency of ABC, however, can be arbitrarily poor for small τ, since a sample X_t is then rarely accepted in Step (iii).

The ABC method generates a sample, hence any statistics based on the posterior can be approximated. Given a posterior mean obtained by one of the kernel methods, however, we may only obtain expectations of functions in the RKHS, meaning that certain statistics (such as confidence intervals) are not straightforward to compute. In Section 5.2, we present an experimental evaluation of the trade-off between computation time and accuracy for ABC and KBR.

5. Experimental Results

This section demonstrates experimental results with the KBR estimator. In addition to the basic comparison with kernel density estimation, we show simple experiments for Bayesian inference without likelihood and filtering with nonparametric hidden Markov models. More practical applications are discussed in other papers; Nishiyama et al. (2012) propose a KBR approach to reinforcement learning in partially observable Markov decision processes, Boots et al. (2013) apply KBR to reinforcement learning with predictive state representations, and Nakagome et al. (2013) consider a KBR-based method for problems in population genetics.

5.1 Nonparametric Inference of Posterior

The first numerical example is a comparison between KBR and a kernel density estimation (KDE) approach to obtaining conditional densities. Let $(X_1, Y_1), \ldots, (X_n, Y_n)$ be an i.i.d. sample from P on $\mathbb{R}^d \times \mathbb{R}^r$. With probability density functions $K^{\mathcal{X}}(x)$ on $\mathbb{R}^d$ and $K^{\mathcal{Y}}(y)$ on $\mathbb{R}^r$, the conditional probability density function $p(y|x)$ is estimated by

$$\widehat{p}(y|x) = \frac{\sum_{j=1}^{n} K^{\mathcal{X}}_{h_X}(x - X_j) K^{\mathcal{Y}}_{h_Y}(y - Y_j)}{\sum_{j=1}^{n} K^{\mathcal{X}}_{h}(x - X_j)},$$

where $K^{\mathcal{X}}_{h_X}(x) = h_X^{-d} K^{\mathcal{X}}(x/h_X)$ and $K^{\mathcal{Y}}_{h_Y}(x) = h_Y^{-r} K^{\mathcal{Y}}(y/h_Y)$ $(h_X, h_Y > 0)$. Given an i.i.d. sample $U_1, \ldots, U_\ell$ from the prior Π, the particle representation of the posterior can be obtained by importance weighting (IW). Using this scheme, the posterior $q(x|y)$ given $y \in \mathbb{R}^r$ is represented by the weighted sample (U_i, ζ_i) with $\zeta_i = \widehat{p}(y|U_i) / \sum_{j=1}^{\ell} \widehat{p}(y|U_j)$.

We compare the estimates of $\int x q(x|y) dx$ obtained by KBR and KDE + IW, using Gaussian kernels for both the methods. We should bear in mind, however, that the function $f(x) = x$ does not

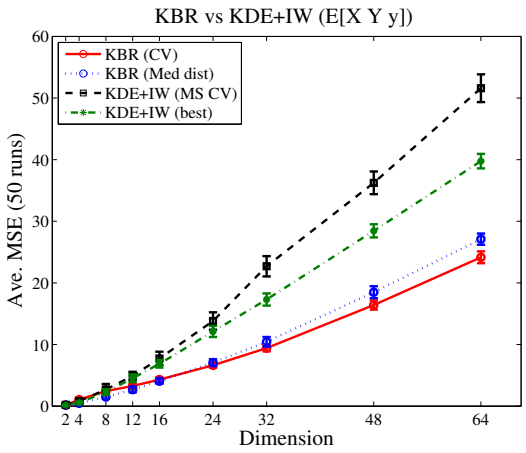


Figure 2: Comparison between KBR and KDE+IW.

belong to the Gaussian kernel RKHS, hence the consistency result of Theorem 6 does not apply to this function. In our experiments, the dimensionality was given by $r = d$ ranging from 2 to 64. The distribution P of (X,Y) was $N((0,\mathbf{1}_d^T)^T, V)$ with $V = A^T A + 2I_d$, where $\mathbf{1}_d = (1,\ldots,1)^T \in \mathbb{R}^d$ and each component of A was randomly generated as $N(0,1)$ for each run. The prior Π was $N(0, V_{XX}/2)$, where V_{XX} is the X-component of V. The sample sizes were $n = \ell = 200$. The bandwidth parameters h_X, h_Y in KDE were set $h_X = h_Y$, and chosen over the set $\{2 * i \mid i = 1,\ldots,10\}$ in two ways: least squares cross-validation (Rudemo, 1982; Bowman, 1984) and the best mean performance. For the KBR, we chose σ in $e^{-\|x-x'\|^2/(2\sigma^2)}$ in two ways: the median over the pairwise distances in the data (Gretton et al., 2008), and the 10-fold cross-validation approach described in Section 4.2. In the latter, σ_x for k_X is first chosen with σ_y for k_Y set as the median distances, and then σ_y is chosen with the best σ_x. Figure 2 shows the mean square errors (MSE) of the estimates over 1000 random points $y \sim N(0, V_{YY})$. KBR significantly outperforms the KDE+IW approach. Unsurprisingly, the MSE of both methods increases with dimensionality.

5.2 Bayesian Computation Without Likelihood

We compare ABC and the kernel methods, KBR and conditional mean, in terms of estimation accuracy and computational time, since they have an obvious tradeoff. To compute the estimation accuracy rigorously, the ground truth is needed: thus we use Gaussian distributions for the true prior and likelihood, which makes the posterior easy to compute in closed form. The samples are taken from the same model used in Section 5.1, and $\int xq(x|y)dx$ is evaluated at 10 different values of y. We performed 10 runs with different randomly chosen parameter values A, representing the "true" distributions.

For ABC, we used only the rejection method; while there are more advanced sampling schemes (Marjoram et al., 2003; Sisson et al., 2007), their implementation is dependent on the problem being solved. Various values for the acceptance region τ are used, and the accuracy and computational time are shown in Fig. 3 together with total sizes of the generated samples. For the kernel methods, the sample size n is varied. The regularization parameters are given by $\varepsilon_n = 0.01/n$ and $\delta_n = 2\varepsilon_n$ for KBR, and $\varepsilon_n = 0.01/\sqrt{n}$ for the conditional kernel mean. The kernels in the kernel methods

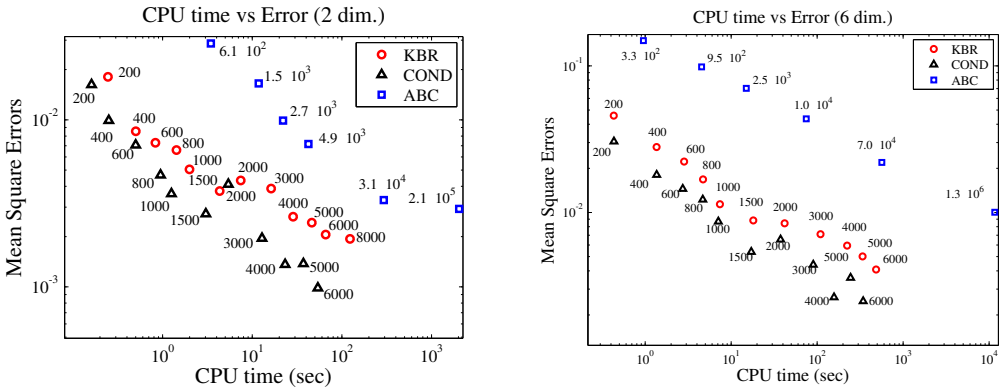


Figure 3: Comparison of estimation accuracy and computational time with KBR and ABC for Bayesian computation without likelihood. COND reprensents the method based on Equation (19). The numbers at the marks are the sample sizes generated for computation.

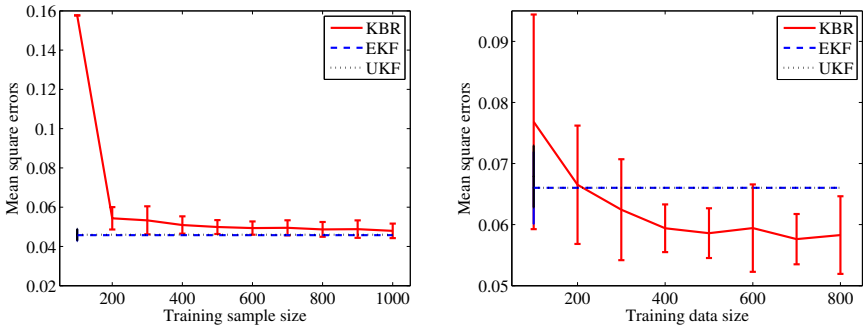
are Gaussian kernels for which the bandwidth parameters are chosen by the median of the pairwise distances on the data (Gretton et al., 2008). The incomplete Cholesky decomposition with tolerance 0.001 is employed for the low-rank approximation. The resulting ranks are, for instance, around 30 for $N = 200$ and around 50 for $N = 6000$ in the case of KBR dimension 2; around 180 for $N = 200$ and 1200 for $N = 6000$ in the case of dimension 6. This implies considerable reduction of computational time especially in the case of large sample sizes. The experimental results indicate that kernel methods achieve more accurate results than ABC at a given computational cost. The conditional kernel mean yields the best results, since in this instance, it is not necessary to correct for a difference in distribution between Π and P_X. In the next experiment, however, this simplification can no longer be made.

5.3 Filtering Problems

We next compare the KBR filtering method (proposed in Section 4.3) with EKF and UKF on synthetic data.

KBR has the regularization parameters ε_T, δ_T, and kernel parameters for k_X and k_Y (e.g., the bandwidth parameter for an RBF kernel). Under the assumption that a training sample is available, cross-validation can be performed on the training sample to select the parameters. By dividing the training sample into two, one half is used to estimate the covariance operators Equation (16) with a candidate parameter set, and the other half to evaluate the estimation errors. To reduce the search space and attendant computational cost, we used a simpler procedure, setting $\delta_T = 2\varepsilon_T$, and Gaussian kernel bandwidths $\beta\sigma_X$ and $\beta\sigma_Y$, where σ_X and σ_Y were the median of pairwise distances in the training samples (Gretton et al., 2008). This left only two parameters β and ε_T to be tuned.

We applied the KBR filtering algorithm from Section 4.3 to two synthetic data sets: a simple nonlinear dynamical system, in which the degree of nonlinearity could be controlled, and the problem of camera orientation recovery from an image sequence. In the first case, the hidden state was



Data (a) Data (b)

Figure 4: Comparisons with the KBR Filter and EKF. (Average MSEs and standard errors over 30 runs.) (a): dynamics with weak nonlinearity (b): dynamics with strong nonlinearity.

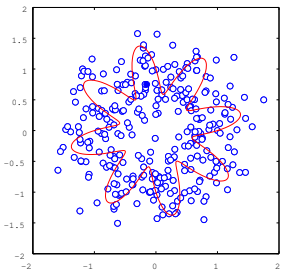


Figure 5: Example of data (b) (X_t, $N = 300$)

$X_t = (u_t, v_t)^T \in \mathbb{R}^2$, and the dynamics were given by

$$\begin{pmatrix} u_{t+1} \\ v_{t+1} \end{pmatrix} = (1 + b\sin(M\theta_{t+1})) \begin{pmatrix} \cos\theta_{t+1} \\ \sin\theta_{t+1} \end{pmatrix} + \zeta_t, \quad \theta_{t+1} = \theta_t + \eta \pmod{2\pi},$$

where $\eta > 0$ is an increment of the angle and $\zeta_t \sim N(0, \sigma_h^2 I_2)$ is independent process noise. Note that the dynamics of (u_t, v_t) were nonlinear even for $b = 0$. The observation Y_t was

$$Y_t = (u_t, v_t)^T + \xi_t, \qquad \xi_t \sim N(0, \sigma_o^2 I),$$

where ξ_t was independent noise. The two dynamics were defined as follows: (a) rotation with noisy observations $\eta = 0.3$, $b = 0$, $\sigma_h = \sigma_o = 0.2$; (b) oscillatory rotation with noisy observations $\eta = 0.4$, $b = 0.4$, $M = 8$, $\sigma_h = \sigma_o = 0.2$. (See Fig.5). We assumed the correct dynamics were known to the EKF and UKF.

Results are shown in Fig. 4. In all cases, the EKF and UKF show an indistinguishably small difference. The dynamics in (a) are weakly nonlinear, and KBR has slightly worse MSE than EKF and UKF. For data set (b), which has strong nonlinearity, KBR outperforms the nonlinear Kalman filter for $T \geq 200$.

In our second synthetic example, we applied the KBR filter to the camera rotation problem used in Song et al. (2009). The angle of a camera, which was located at a fixed position, was a

	KBR (Gauss)	KBR (Tr)	Kalman (9 dim.)	Kalman (Quat.)
$\sigma^2 = 10^{-4}$	0.210 ± 0.015	0.146 ± 0.003	1.980 ± 0.083	0.557 ± 0.023
$\sigma^2 = 10^{-3}$	0.222 ± 0.009	0.210 ± 0.008	1.935 ± 0.064	0.541 ± 0.022

Table 1: Average MSE and standard errors for camera angle estimation (10 runs).

hidden variable, and movie frames recorded by the camera were observed. The data were generated virtually using a computer graphics environment. As in Song et al. (2009), we were given 3600 downsampled frames of 20×20 RGB pixels ($Y_t \in [0,1]^{1200}$), where the first 1800 frames were used for training, and the second half were used to test the filter. We made the data noisy by adding Gaussian noise $N(0, \sigma^2)$ to Y_t.

Our experiments covered two settings. In the first, we did not use that the hidden state S_t was included in $SO(3)$, but only that it was a general 3×3 matrix. In this case, we formulated the Kalman filter by estimating the relations under a linear assumption, and the KBR filter with Gaussian kernels for S_t and X_t as Euclidean vectors. In the second setting, we exploited the fact that $S_t \in SO(3)$: for the Kalman filter, S_t was represented by a quanternion, which is a standard vector representation of rotations; for the KBR filter the kernel $k(A,B) = \mathrm{Tr}[AB^T]$ was used for S_t, and S_t was estimated within $SO(3)$. Table 1 shows the Frobenius norms between the estimated matrix and the true one. The KBR filter significantly outperforms the EKF, since KBR is able to extract the complex nonlinear dependence between the observation and the hidden state.

6. Proofs

This section includes the proofs of the theoretical results in Section 3.2. The proof ideas are similar to Caponnetto and De Vito (2007) and Smale and Zhou (2007), in which the basic techniques are taken from the general theory of regularization (Engl et al., 2000). Before stating the main proofs, we provide a proof of consistency of the empirical counterparts in Proposition 3 to the kernel Sum rule (Theorem 2). We then proceed to the proofs of Theorems 5 and 6, covering the consistency of the KBR procedure in RKHS, followed by a proof of Theorem 7 for consistency in L^2.

6.1 Consistency of the Kernel Sum Rule

We show consistency of an empirical estimate of $m_{Q_\mathcal{Y}}$ in Theorem 2. The same proof also applies in establishing consistency for the empirical estimates $\widehat{C}^{(n)}_{WW}$ and $\widehat{C}^{(n)}_{WZ}$ in Proposition 3.

Theorem 8 *Assume that C_{XX} is injective, $\widehat{m}_\Pi$ is a consistent estimator of m_Π in $\mathcal{H}_\mathcal{X}$ norm, and that $E[k_\mathcal{Y}(Y,\tilde{Y})|X = x, \tilde{X} = \tilde{x}]$ is included in $\mathcal{H}_\mathcal{X} \otimes \mathcal{H}_\mathcal{X}$ as a function of $(x,\tilde{x})$, where $(\tilde{X},\tilde{Y})$ is an independent copy of (X,Y). Then, if the regularization coefficient ε_n decays to zero sufficiently slowly,*

$$\left\| \widehat{C}^{(n)}_{YX} \left(\widehat{C}^{(n)}_{XX} + \varepsilon_n I \right)^{-1} \widehat{m}_\Pi - m_{Q_\mathcal{Y}} \right\|_{\mathcal{H}_\mathcal{Y}} \to 0$$

in probability as $n \to \infty$.

Proof The assertion is proved if, as $n \to \infty$,

$$\| \widehat{C}^{(n)}_{YX} (\widehat{C}^{(n)}_{XX} + \varepsilon_n I)^{-1} \widehat{m}_\Pi - C_{YX} (C_{XX} + \varepsilon_n I)^{-1} m_\Pi \|_{\mathcal{H}_\mathcal{Y}} \to 0 \tag{20}$$

in probability and

$$\|C_{YX}(C_{XX}+\varepsilon_n I)^{-1}m_\Pi - m_{Q_\mathcal{Y}}\|_{\mathcal{H}_\mathcal{Y}} \to 0 \tag{21}$$

with an appropriate choice of ε_n.

By using the fact that $B^{-1}-A^{-1}=B^{-1}(A-B)A^{-1}$ holds for any invertible operators A and B, the left hand side of Equation (20) is upper bounded by

$$\begin{aligned}\left\|\widehat{C}_{YX}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\left(\widehat{m}_\Pi - m_\Pi\right)\right\|_{\mathcal{H}_\mathcal{Y}} + \left\|\left(\widehat{C}_{YX}^{(n)}-C_{YX}\right)\left(C_{XX}+\varepsilon_n I\right)^{-1}m_\Pi\right\|_{\mathcal{H}_\mathcal{Y}} \\ + \left\|\widehat{C}_{YX}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\left(C_{XX}-\widehat{C}_{XX}^{(n)}\right)\left(C_{XX}+\varepsilon_n I\right)^{-1}m_\Pi\right\|_{\mathcal{H}_\mathcal{Y}}. \end{aligned}\tag{22}$$

By the decomposition

$$\widehat{C}_{YX}^{(n)} = \widehat{C}_{YY}^{(n)1/2}\widehat{W}_{YX}^{(n)}\widehat{C}_{XX}^{(n)1/2}$$

with $\|\widehat{W}_{YX}^{(n)}\| \le 1$ (Baker, 1973), we have

$$\|\widehat{C}_{YX}^{(n)}(\widehat{C}_{XX}^{(n)}+\varepsilon_n I)^{-1}\| \le \|\widehat{C}_{YY}^{(n)1/2}\widehat{W}_{YX}^{(n)}(\widehat{C}_{XX}^{(n)}+\varepsilon_n I)^{-1/2}\| = O_p(\varepsilon_n^{-1/2}),$$

which implies the first term is of $O_p(\varepsilon_n^{-1/2}\|\widehat{m}_\Pi - m_\Pi\|_{\mathcal{H}_\mathcal{X}})$. From the $\sqrt{n}$-consistency of the covariance operators, the second and third terms are of the order $O_p(n^{-1/2}\varepsilon_n^{-1})$ and $O_p(n^{-1/2}\varepsilon_n^{-3/2})$, respectively. If ε_n is taken so that $\varepsilon_n \gg n^{-1/3}$ and $\varepsilon_n \gg \|\widehat{m}_\Pi^{(n)} - m_\Pi\|^2_{\mathcal{H}_\mathcal{X}}$, Equation (20) converges to zero in probability.

For Equation (21), first note

$$\begin{aligned}&\|C_{YX}(C_{XX}+\varepsilon_n I)^{-1}m_\Pi - m_{Q_\mathcal{Y}}\|^2_{\mathcal{H}_\mathcal{Y}} \\ &\quad = \|C_{YX}(C_{XX}+\varepsilon_n I)^{-1}m_\Pi\|^2_{\mathcal{H}_\mathcal{X}} - 2\langle C_{YX}(C_{XX}+\varepsilon_n I)^{-1}m_\Pi, m_{Q_\mathcal{Y}}\rangle_{\mathcal{H}_\mathcal{Y}} + \|m_{Q_\mathcal{Y}}\|^2_{\mathcal{H}_\mathcal{Y}}.\end{aligned}$$

Let $\theta(x,\tilde{x}) := E[k_\mathcal{Y}(Y,\tilde{Y})|X=x,\tilde{X}=\tilde{x}]$. The third term in the right hand side is

$$\begin{aligned}\langle m_{Q_\mathcal{Y}}, m_{Q_\mathcal{Y}}\rangle_{\mathcal{H}_\mathcal{Y}} &= \int\int m_{Q_\mathcal{Y}}(y)dP_{\mathcal{Y}|x}(y)d\Pi(x) \\ &= \int\int \langle m_{Q_\mathcal{Y}}, k_\mathcal{Y}(\cdot,y)\rangle_{\mathcal{H}_\mathcal{Y}} dP_{\mathcal{Y}|x}(y)d\Pi(x) = \int\int \theta(x,\tilde{x})d\Pi(x)d\Pi(\tilde{x}).\end{aligned}$$

From the assumption $\theta \in \mathcal{H}_\mathcal{X}\otimes\mathcal{H}_\mathcal{X}$ and the fact $E[m_{Q_\mathcal{Y}}(Y)|X=\cdot] = \int\theta(\cdot,\tilde{x})d\Pi(\tilde{x})$, Lemma 9 below shows $E[m_{Q_\mathcal{Y}}(Y)|X=\cdot]\in\mathcal{H}_\mathcal{X}$. It follows from Theorem 1 that $C_{XY}m_{Q_\mathcal{Y}} = C_{XX}E[m_{Q_\mathcal{Y}}(Y)|X=\cdot]$ and thus

$$\begin{aligned}\langle C_{YX}(C_{XX}+\varepsilon_n I)^{-1}m_\Pi, m_{Q_\mathcal{Y}}\rangle_{\mathcal{H}_\mathcal{Y}} &= \langle m_\Pi, (C_{XX}+\varepsilon_n I)^{-1}C_{XY}m_{Q_\mathcal{Y}}\rangle_{\mathcal{H}_\mathcal{X}} \\ &= \langle m_\Pi, (C_{XX}+\varepsilon_n I)^{-1}C_{XX}E[m_{Q_\mathcal{Y}}(Y)|X=\cdot]\rangle_{\mathcal{H}_\mathcal{X}},\end{aligned}$$

which converges to

$$\langle m_\Pi, E[m_{Q_\mathcal{Y}}(Y)|X=\cdot]\rangle_{\mathcal{H}_\mathcal{X}} = \int\int\theta(x,\tilde{x})d\Pi(x)d\Pi(\tilde{x})$$

from Lemma 10 below.

Note that

$$\|C_{YX}f\|^2_{\mathcal{H}_\mathcal{Y}} = \langle C_{YX}f, C_{YX}f\rangle_{\mathcal{H}_\mathcal{X}} = E[f(X)(C_{YX}f)(Y)] \\ = E[f(X)E[k_\mathcal{Y}(Y,\tilde{Y})f(\tilde{X})]] = E[f(X)f(\tilde{X})\theta(X,\tilde{X})]$$

for any $f \in \mathcal{H}_\mathcal{X}$, where $(\tilde{X},\tilde{Y})$ is an independent copy of (X,Y). By taking $f = (C_{XX}+\varepsilon_n I)^{-1}m_\Pi$, the first term is given by

$$\begin{aligned} &E\left[\theta(X,\tilde{X})((C_{XX}+\varepsilon_n I)^{-1}m_\Pi)(X)((C_{XX}+\varepsilon_n I)^{-1}m_\Pi)(\tilde{X})\right] \\ &= E\left[\theta(X,\tilde{X})f(X)f(\tilde{X})\right] \\ &= \left\langle \theta, (C_{XX}\otimes C_{XX})f\otimes f)\right\rangle_{\mathcal{H}_\mathcal{X}\otimes\mathcal{H}_\mathcal{X}} \\ &= \left\langle \theta, (C_{XX}(C_{XX}+\varepsilon_n I)^{-1}m_\Pi)\otimes(C_{XX}(C_{XX}+\varepsilon_n I)^{-1}m_\Pi)\right\rangle_{\mathcal{H}_\mathcal{X}\otimes\mathcal{H}_\mathcal{X}}, \end{aligned}$$

which, by Lemma 10, converges to

$$\langle \theta, m_\Pi\otimes m_\Pi\rangle_{\mathcal{H}_\mathcal{X}\otimes\mathcal{H}_\mathcal{X}} = \int\int \theta(x,\tilde{x})d\Pi(x)d\Pi(\tilde{x}).$$

This completes the proof. ■

Lemma 9 *Let $\mathcal{H}_\mathcal{X}$ and $\mathcal{H}_\mathcal{Y}$ be RKHS's on $\mathcal{X}$ and $\mathcal{Y}$, respectively. If a function θ on $\mathcal{X}\times\mathcal{Y}$ is in $\mathcal{H}_\mathcal{X}\otimes\mathcal{H}_\mathcal{Y}$ (product space), then for any fixed $y\in\mathcal{Y}$ the function $\theta(\cdot,y)$ of the first argument is in $\mathcal{H}_\mathcal{X}$.*

Proof Let $\{\phi_i\}_{i=1}^I$ and $\{\psi_j\}_{j=1}^J$ be complete orthonormal bases of $\mathcal{H}_\mathcal{X}$ and $\mathcal{H}_\mathcal{Y}$, respectively, where $I,J\in\mathbb{N}\cup\{\infty\}$. Then θ is expressed as

$$\theta = \sum_{i=1}^I\sum_{j=1}^J \alpha_{ij}\phi_i\psi_j$$

with $\sum_{i,j}|\alpha_{ij}|^2<\infty$ (e.g., Aronszajn, 1950). We have $\theta(\cdot,y)=\sum_{i=1}^I\beta_i\phi_i$ with $\beta_i=\sum_{j=1}^J\alpha_{ij}\psi_j(y)$. Since

$$\begin{aligned} &\sum_i|\beta_i|^2 = \sum_i\Big|\sum_j\alpha_{ij}\psi_j(y)\Big|^2 \\ &\le \sum_i\sum_j|\alpha_{ij}|^2\sum_j|\psi_j(y)|^2 = \sum_{ij}|\alpha_{ij}|^2\sum_j\langle\psi_j,k_\mathcal{Y}(\cdot,y)\rangle^2_{\mathcal{H}_\mathcal{Y}} = \sum_{ij}|\alpha_{ij}|^2\|k_\mathcal{Y}(\cdot,y)\|^2_{\mathcal{H}_\mathcal{Y}}<\infty, \end{aligned}$$

we have $\theta(\cdot,y)\in\mathcal{H}_\mathcal{X}$. ■

Lemma 10 *Let $\mathcal{H}$ be a separable Hilbert space and C be a positive, injective, self-adjoint, compact operator on $\mathcal{H}$. then, for any $f\in\mathcal{H}$,*

$$((C+\varepsilon I)^{-1}Cf\to f,\qquad(\varepsilon\to+0).$$

Proof By the assumptions, there exist an orthonormal basis $\{\phi_i\}$ for $\mathcal{H}$ and positive eigenvalues λ_i such that

$$Cf = \sum_i \lambda_i \langle f, \phi_i \rangle_{\mathcal{H}} \phi_i.$$

Then, we have

$$\|(C+\varepsilon I)^{-1} Cf - f\|_{\mathcal{H}}^2 = \sum_i \left| \frac{\varepsilon}{\lambda_i + \varepsilon} \right|^2 |\langle f, \phi_i \rangle_{\mathcal{H}}|^2.$$

Since $\left|(\varepsilon/\lambda_i + \varepsilon)\right|^2 \leq 1$ and $\sum_i |\langle f, \phi_i \rangle_{\mathcal{H}}|^2 = \|f\|_{\mathcal{H}}^2 < \infty$, the dominated convergence theorem ensures

$$\lim_{\varepsilon \to +0} \|(C+\varepsilon I)^{-1} Cf - f\|_{\mathcal{H}}^2 = \sum_i \lim_{\varepsilon \to +0} \left| \frac{\varepsilon}{\lambda_i + \varepsilon} \right|^2 |\langle f, \phi_i \rangle_{\mathcal{H}}|^2 = 0,$$

which completes the proof. ■

6.2 Consistency Results in RKHS Norm

We first prove Theorem 5.

Proof [Proof of Theorem 5] By replacing $Y \mapsto (X,Y)$ and $Y \mapsto (Y,Y)$, it follows from Theorem 8 that $\widehat{C}_{ZW}^{(n)}$ and $\widehat{C}_{WW}^{(n)}$ are consistent estimators of C_{ZW} and C_{WW}, respectively, in operator norm. For the proof, it then suffices to show

$$\left\| \widehat{C}_{ZW}^{(n)} \left(\left(\widehat{C}_{WW}^{(n)} \right)^2 + \delta_n I \right)^{-1} \widehat{C}_{WW}^{(n)} k_{\mathcal{Y}}(\cdot, y) - C_{ZW} (C_{WW}^2 + \delta_n I)^{-1} C_{WW} k_{\mathcal{Y}}(\cdot, y) \right\|_{\mathcal{H}_{\mathcal{X}}} \to 0$$

in probability and

$$\left\| C_{ZW} (C_{WW}^2 + \delta_n I)^{-1} C_{WW} k_{\mathcal{Y}}(\cdot, y) - m_{Q_{X|y}} \right\|_{\mathcal{H}_{\mathcal{X}}} \to 0$$

with an appropriate choice of δ_n. The proof of the first convergence is similar to the proof of Equation (20) in Theorem 8, and we omit it. The proof of the second convergence is also similar to Theorem 8. The square of the left hand side is decomposed as

$$\begin{aligned} &\left\| C_{ZW} (C_{WW}^2 + \delta_n I)^{-1} C_{WW} k_{\mathcal{Y}}(\cdot, y) \right\|_{\mathcal{H}_{\mathcal{X}}}^2 \\ &\qquad - 2 \left\langle C_{ZW} (C_{WW}^2 + \delta_n I)^{-1} C_{WW} k_{\mathcal{Y}}(\cdot, y), m_{Q_{X|y}} \right\rangle_{\mathcal{H}_{\mathcal{X}}} + \left\| m_{Q_{X|y}} \right\|_{\mathcal{H}_{\mathcal{X}}}^2. \end{aligned}$$

Let $\xi \in \mathcal{H}_{\mathcal{Y}} \otimes \mathcal{H}_{\mathcal{Y}}$ be defined by $\xi(y, \tilde{y}) := E[k_{\mathcal{X}}(Z, \tilde{Z}) | W = y, \tilde{W} = \tilde{y}]$, where $(\tilde{Z}, \tilde{W})$ be an independent copy of (Z, W). The third term is then equal to

$$\xi(y, y) = E[k_{\mathcal{X}}(Z, \tilde{Z}) | W = y, \tilde{W} = y].$$

For the second term, by the same argument as the proof of Theorem 8, we have

$$\xi(y, \cdot) = E[k_{\mathcal{X}}(Z, \tilde{Z}) | W = y, \tilde{W} = \cdot] \in \mathcal{H}_{\mathcal{Y}}$$

via Lemma 9, and

$$C_{WZ} m_{Q_{X|y}} = C_{WW} E[k_{\mathcal{X}}(Z, \tilde{Z}) | W = \cdot, \tilde{W} = y] = C_{WW} \xi(\cdot, y) \in \mathcal{H}_{\mathcal{Y}}.$$

We then obtain

$$\left\langle C_{ZW}(C_{WW}^2+\delta_n I)^{-1}C_{WW}k_{\mathcal{Y}}(\cdot,y), m_{Q_{X|y}}\right\rangle_{\mathcal{H}_X} = \left\langle k_{\mathcal{Y}}(\cdot,y), (C_{WW}^2+\delta_n I)^{-1}C_{WW}^2\xi(\cdot,y)\right\rangle_{\mathcal{H}_X},$$

which converges to $\xi(y,y)$.

Finally, defining $\varphi_\delta := (C_{WW}^2+\delta I)^{-1}C_{WW}k_{\mathcal{Y}}(\cdot,y)$, the first term is equal to

$$\begin{aligned}
&E\left[\varphi_\delta(W)\varphi_\delta(\tilde{W})E[k_{\mathcal{X}}(Z,\tilde{Z})|W,\tilde{W}]\right] = \left\langle C_{ZW}\varphi_\delta, C_{ZW}\varphi_\delta\right\rangle_{\mathcal{H}_X} = E\left[\varphi_\delta(W)\left[C_{ZW}\varphi_\delta\right](Z)\right]\\
&= E\left[k(Z,Z')\varphi_\delta(W)\varphi_\delta(\tilde{W})\right] = E\left[\xi(W,\tilde{W})\varphi_\delta(W)\varphi_\delta(\tilde{W})\right]\\
&= \left\langle (C_{WW}\otimes C_{WW})\varphi_\delta\otimes\varphi_\delta, \xi\right\rangle_{\mathcal{H}_{\mathcal{Y}}\otimes\mathcal{H}_{\mathcal{Y}}} = \left\langle (C_{WW}\varphi_\delta)\otimes(C_{WW}\varphi_\delta), \xi\right\rangle_{\mathcal{H}_{\mathcal{Y}}\otimes\mathcal{H}_{\mathcal{Y}}},
\end{aligned}$$

which converges to $\langle k_{\mathcal{Y}}(\cdot,y)\otimes k_{\mathcal{Y}}(\cdot,y), \xi\rangle_{\mathcal{H}_{\mathcal{Y}}\otimes\mathcal{H}_{\mathcal{Y}}} = \xi(y,y)$. This completes the proof. ■

We next show the convergence rate of expectation (Theorem 6) under stronger assumptions. The first result is a rate of convergence for the mean transition in Theorem 2. In the following, $\mathcal{R}(C_{XX}^0)$ means $\mathcal{H}_X$.

Theorem 11 *Assume that the Radon-Nikodym derivative $d\Pi/dP_X$ is included in $\mathcal{R}(C_{XX}^\beta)$ for some $\beta \geq 0$, and let $\widehat{m}_\Pi^{(n)}$ be an estimator of m_Π such that $\|\widehat{m}_\Pi^{(n)} - m_\Pi\|_{\mathcal{H}_X} = O_p(n^{-\alpha})$ as $n\to\infty$ for some $0<\alpha\leq 1/2$. Then, with $\varepsilon_n = n^{-\max\{\frac{2}{3}\alpha, \frac{\alpha}{1+\beta}\}}$, we have*

$$\left\|\widehat{C}_{YX}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\widehat{m}_\Pi^{(n)} - m_{Q_{\mathcal{Y}}}\right\|_{\mathcal{H}_{\mathcal{Y}}} = O_p(n^{-\min\{\frac{2}{3}\alpha, \frac{2\beta+1}{2\beta+2}\alpha\}}), \quad (n\to\infty).$$

Proof Take $\eta\in\mathcal{H}_X$ such that $d\Pi/dP_X = C_{XX}^\beta\eta$. Then, we have

$$m_\Pi = \int k_{\mathcal{X}}(\cdot,x)\left(\frac{d\Pi}{dP_{\mathcal{X}}}\right)(x)dP_{\mathcal{X}}(x) = C_{XX}^{\beta+1}\eta. \tag{23}$$

First we show the rate of the estimation error:

$$\left\|\widehat{C}_{YX}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\widehat{m}_\Pi^{(n)} - C_{YX}\left(C_{XX}+\varepsilon_n I\right)^{-1}m_\Pi\right\|_{\mathcal{H}_{\mathcal{Y}}} = O_p\left(n^{-\alpha}\varepsilon_n^{-1/2}\right), \tag{24}$$

as $n\to\infty$. The left hand side of Equation (24) is upper bounded by

$$\begin{aligned}
&\left\|\widehat{C}_{YX}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\left(\widehat{m}_\Pi^{(n)}-m_\Pi\right)\right\|_{\mathcal{H}_{\mathcal{Y}}} + \left\|\left(\widehat{C}_{YX}^{(n)}-C_{YX}\right)\left(C_{XX}+\varepsilon_n I\right)^{-1}m_\Pi\right\|_{\mathcal{H}_{\mathcal{Y}}}\\
&\qquad + \left\|\widehat{C}_{YX}^{(n)}\left(\widehat{C}_{XX}^{(n)}+\varepsilon_n I\right)^{-1}\left(C_{XX}-\widehat{C}_{XX}^{(n)}\right)\left(C_{XX}+\varepsilon_n I\right)^{-1}m_\Pi\right\|_{\mathcal{H}_{\mathcal{Y}}}.
\end{aligned}$$

In a similar manner to derivation of the bound of Equation (22), we obtain Equation (24).

Next, we show the rate for the approximation error

$$\left\|C_{YX}\left(C_{XX}+\varepsilon_n I\right)^{-1}m_\Pi - m_{Q_{\mathcal{Y}}}\right\|_{\mathcal{H}_{\mathcal{Y}}} = O(\varepsilon_n^{\min\{(1+2\beta)/2,1\}}) \qquad (n\to\infty). \tag{25}$$

Let $C_{YX} = C_{YY}^{1/2}W_{YX}C_{XX}^{1/2}$ be the decomposition with $\|W_{YX}\|\leq 1$. It follows from Equation (23) and the relation

$$\begin{aligned}
m_{Q_{\mathcal{Y}}} = \int\int k_{\mathcal{Y}}(\cdot,y)dP_{\mathcal{Y}|x}(y)d\Pi(x) &= \int\int k(\cdot,y)\left(\frac{d\Pi}{dP_{\mathcal{X}}}\right)(x)dP_{\mathcal{Y}|x}(y)dP_{\mathcal{X}}(x)\\
&= \int\int k(\cdot,y)\left(\frac{d\Pi}{dP_{\mathcal{X}}}\right)(x)dP(x,y) = C_{YX}C_{XX}^\beta\eta
\end{aligned}$$

that the left hand side of Equation (25) is upper bounded by

$$\|C_{YY}^{1/2}W_{YX}\|\,\|\big(C_{XX}+\varepsilon_n I\big)^{-1}C_{XX}^{(2\beta+3)/2}\eta - C_{XX}^{(2\beta+1)/2}\eta\|_{\mathcal{H}_X} \\ \leq \varepsilon_n\|C_{YY}^{1/2}W_{YX}\|\,\|(C_{XX}+\varepsilon_n I)^{-1}C_{XX}^{\beta+1/2}\|\,\|\eta\|_{\mathcal{H}_X}.$$

For $\beta \geq 1/2$, it follows from

$$\varepsilon_n\|(C_{XX}+\varepsilon_n I)^{-1}C_{XX}^{\beta+1/2}\| \leq \varepsilon_n\|(C_{XX}+\varepsilon_n I)^{-1}C_{XX}\|\,\|C_{XX}^{\beta-1/2}\|$$

that the left hand side of Equation (25) converges to zero in $O(\varepsilon_n)$. If $0 \leq \beta < 1/2$, we have

$$\varepsilon_n\|(C_{XX}+\varepsilon_n I)^{-1}C_{XX}^{\beta+1/2}\| \\ = \varepsilon_n^{\beta+1/2}\|\varepsilon_n^{1/2-\beta}(C_{XX}+\varepsilon_n I)^{-(1/2-\beta)}\|\,\|(C_{XX}+\varepsilon_n I)^{-\beta-1/2}C_{XX}^{\beta+1/2}\| \leq \varepsilon_n^{\beta+1/2},$$

which proves Equation (25).

With the order of ε_n to balance Equations (24) and (25), the asserted rate of consistency is obtained. ■

The following theorem shows the convergence rate of the estimator used in the second step of KBR.

Theorem 12 *Let f be a function in $\mathcal{H}_X$, and (Z,W) be a random variable taking values in $\mathcal{X}\times\mathcal{Y}$. Assume that $E[f(Z)|W=\cdot] \in \mathcal{R}(C_{WW}^{\nu})$ for some $\nu \geq 0$, and that $\widehat{C}_{WZ}^{(n)} : \mathcal{H}_X \to \mathcal{H}_Y$ and $\widehat{C}_{WW}^{(n)} : \mathcal{H}_Y \to \mathcal{H}_Y$ are bounded operators such that $\|\widehat{C}_{WZ}^{(n)} - C_{WZ}\| = O_p(n^{-\gamma})$ and $\|\widehat{C}_{WW}^{(n)} - C_{WW}\| = O_p(n^{-\gamma})$ for some $\gamma > 0$. Then, for a positive sequence $\delta_n = n^{-\max\{\frac{4}{9}\gamma, \frac{4}{2\nu+5}\gamma\}}$, we have as $n\to\infty$*

$$\big\|\widehat{C}_{WW}^{(n)}\big((\widehat{C}_{WW}^{(n)})^2+\delta_n I\big)^{-1}\widehat{C}_{WZ}^{(n)}f - E[f(Z)|W=\cdot]\big\|_{\mathcal{H}_Y} = O_p(n^{-\min\{\frac{4}{9}\gamma, \frac{2\nu}{2\nu+5}\gamma\}}).$$

Proof Let $\eta \in \mathcal{H}_X$ such that $E[f(Z)|W=\cdot] = C_{WW}^{\nu}\eta$. First we show

$$\big\|\widehat{C}_{WW}^{(n)}\big((\widehat{C}_{WW}^{(n)})^2+\delta_n I\big)^{-1}\widehat{C}_{WZ}^{(n)}f - C_{WW}(C_{WW}^2+\delta_n I)^{-1}C_{WZ}f\big\|_{\mathcal{H}_Y} = O_p(n^{-\gamma}\delta_n^{-5/4}). \qquad (26)$$

The left hand side of Equation (26) is upper bounded by

$$\big\|\widehat{C}_{WW}^{(n)}\big((\widehat{C}_{WW}^{(n)})^2+\delta_n I\big)^{-1}(\widehat{C}_{WZ}^{(n)}-C_{WZ})f\big\|_{\mathcal{H}_Y} \\ + \big\|(\widehat{C}_{WW}^{(n)}-C_{WW})(C_{WW}^2+\delta_n I)^{-1}C_{WZ}f\big\|_{\mathcal{H}_Y} \\ + \big\|\widehat{C}_{WW}^{(n)}((\widehat{C}_{WW}^{(n)})^2+\delta_n I)^{-1}\big((\widehat{C}_{WW}^{(n)})^2-C_{WW}^2\big)\big(C_{WW}^2+\delta_n I\big)^{-1}C_{WZ}f\big\|_{\mathcal{H}_Y}.$$

For $A = \widehat{C}_{WW}^{(n)}$, we have

$$\|A(A^2+\delta_n I)^{-1}\| = \|\{A^2(A^2+\delta_n I)^{-1}\}^{1/2}(A^2+\delta_n I)^{-1/2}\| \leq \delta_n^{-1/2},$$

and thus the first term of the above bound is of $O_p(n^{-\gamma}\delta_n^{-1/2})$. A similar argument to C_{WW} combined with the decomposition $C_{WZ} = C_{WW}^{1/2} U_{WZ} C_{ZZ}^{1/2}$ with $\|U_{WZ}\| \leq 1$ shows that the second term is of $O_p(n^{-\gamma}\delta_n^{-3/4})$. From the fact

$$\|(\widehat{C}_{WW}^{(n)})^2 - C_{WW}^2\| \leq \|\widehat{C}_{WW}^{(n)}(\widehat{C}_{WW}^{(n)} - C_{WW})\| + \|(\widehat{C}_{WW}^{(n)} - C_{WW})C_{WW}\| = O_p(n^{-\gamma}),$$

the third term is of $O_p(n^{-\gamma}\delta_n^{-5/4})$. This implies Equation (26).

From $E[f(Z)|W=\cdot] = C_{WW}^{\nu}\eta$ and $C_{WZ}f = C_{WW}E[f(Z)|W=\cdot] = C_{WW}^{\nu+1}\eta$, the convergence rate

$$\left\|C_{WW}(C_{WW}^2 + \delta_n I)^{-1} C_{WZ} f - E[f(Z)|W=\cdot]\right\|_{\mathcal{H}_\mathcal{Y}} = O(\delta_n^{\min\{1,\frac{\nu}{2}\}}). \tag{27}$$

can be proved in the same way as Equation (25).

Finally, combining Equations (26) and (27) proves the assertion. ■

6.3 Consistency Results in L^2

Recall that $\tilde{C}_{WW}$ is the integral operator on $L^2(Q_\mathcal{Y})$ defined by Equation (15). The following theorem shows the convergence rate on average. Here $\mathcal{R}(\tilde{C}_{WW}^0)$ means $L^2(Q_\mathcal{Y})$. In the following the canonical mapping from $\mathcal{H}_\mathcal{Y}$ to $L^2(Q_\mathcal{Y})$ is denoted by $J_\mathcal{Y}$. The mapping $J_\mathcal{X} : \mathcal{H}_\mathcal{X} \to L^2(Q_\mathcal{X})$ is defined similarly.

Theorem 13 *Let f be a function in $\mathcal{H}_\mathcal{X}$, and (Z,W) be a random variable taking values in $\mathcal{X}\times\mathcal{Y}$ with distribution Q. Assume that $E[f(Z)|W=\cdot] \in \mathcal{H}_\mathcal{Y}$ and $J_\mathcal{Y}E[f(Z)|W=\cdot] \in \mathcal{R}(\tilde{C}_{WW}^{\nu})$ for some $\nu > 0$, and that $\widehat{C}_{WZ}^{(n)} : \mathcal{H}_\mathcal{X} \to \mathcal{H}_\mathcal{Y}$ and $\widehat{C}_{WW}^{(n)} : \mathcal{H}_\mathcal{Y} \to \mathcal{H}_\mathcal{Y}$ are bounded operators such that $\|\widehat{C}_{WZ}^{(n)} - C_{WZ}\| = O_p(n^{-\gamma})$ and $\|\widehat{C}_{WW}^{(n)} - C_{WW}\| = O_p(n^{-\gamma})$ for some $\gamma > 0$. Then, for a positive sequence $\delta_n = n^{-\max\{\frac{1}{2}\gamma, \frac{2}{\nu+2}\gamma\}}$, we have as $n \to \infty$*

$$\left\|J_\mathcal{Y}\widehat{C}_{WW}^{(n)}\left((\widehat{C}_{WW}^{(n)})^2 + \delta_n I\right)^{-1}\widehat{C}_{WZ}^{(n)} f - J_\mathcal{Y}E[f(Z)|W=\cdot]\right\|_{L^2(Q_\mathcal{Y})} = O_p(n^{-\min\{\frac{1}{2}\gamma, \frac{\nu}{\nu+2}\gamma\}}),$$

where $Q_\mathcal{Y}$ is the marginal distribution of W.

Proof Note that for $f, g \in \mathcal{H}_\mathcal{X}$ we have $(Jf, Jg)_{L^2(Q_\mathcal{Y})} = E[f(W)g(W)] = \langle f, C_{WW}g\rangle_{\mathcal{H}_\mathcal{X}}$. It follows that the left hand side of the assertion is equal to

$$\left\|C_{WW}^{1/2}\left\{\widehat{C}_{WW}^{(n)}\left((\widehat{C}_{WW}^{(n)})^2 + \delta_n I\right)^{-1}\widehat{C}_{WZ}^{(n)} f - E[f(Z)|W=\cdot]\right\}\right\|_{\mathcal{H}_\mathcal{Y}}.$$

First, by a similar argument to the proof of Equation (26), it is easy to show that the rate of the estimation error is given by

$$\left\|C_{WW}^{1/2}\left\{\widehat{C}_{WW}^{(n)}\left((\widehat{C}_{WW}^{(n)})^2 + \delta_n I\right)^{-1}\widehat{C}_{WZ}^{(n)} f - C_{WW}(C_{WW}^2 + \delta_n I)^{-1}C_{WZ}f\right\}\right\|_{\mathcal{H}_\mathcal{Y}} = O_p(n^{-\gamma}\delta_n^{-1}).$$

It suffices then to prove

$$\left\|J_\mathcal{Y}C_{WW}(C_{WW}^2 + \delta_n I)^{-1}C_{WZ}f - J_\mathcal{Y}E[f(Z)|W=\cdot]\right\|_{L^2(Q_\mathcal{Y})} = O(\delta_n^{\min\{1,\frac{\nu}{2}\}}).$$

Let $\xi \in L^2(Q_{\mathcal{Y}})$ such that $J_{\mathcal{Y}} E[f(Z)|W = \cdot] = \tilde{C}_{WW}^{\nu} \xi$. In a similar way to Theorem 1, $\tilde{C}_{WW} J_{\mathcal{Y}} E[f(Z)|W = \cdot] = \tilde{C}_{WZ} J_{\mathcal{X}} f$ holds, where $\tilde{C}_{WZ}$ is the extension of C_{WZ} to an operator from $L^2(Q_{\mathcal{X}})$ to $L^2(Q_{\mathcal{Y}})$, and thus $J_{\mathcal{Y}} C_{WZ} f = \tilde{C}_{WW}^{\nu+1} \xi$. If follows from $J_{\mathcal{Y}} C_{WW} = \tilde{C}_{WW} J_{\mathcal{Y}}$ that the left hand side of the above equation is equal to

$$\left\| \tilde{C}_{WW} (\tilde{C}_{WW}^2 + \delta_n I)^{-1} \tilde{C}_{WW}^{\nu+1} \xi - \tilde{C}_{WW}^{\nu} \xi \right\|_{L^2(Q_{\mathcal{Y}})}.$$

A similar argument to the proof of Equation (27) shows the assertion. ■

The convergence rate of KBR follows by combining the above theorems.

Theorem 14 *Let f be a function in $\mathcal{H}_{\mathcal{X}}$, (Z,W) be a random variable that has the distribution Q defined by Equation (5), and $\widehat{m}_{\Pi}^{(n)}$ be an estimator of m_{Π} such that $\|\widehat{m}_{\Pi}^{(n)} - m_{\Pi}\|_{\mathcal{H}_{\mathcal{X}}} = O_p(n^{-\alpha})$ $(n \to \infty)$ for some $0 < \alpha \leq 1/2$. Assume that the Radon Nikodym derivative $d\Pi/dP_{\mathcal{X}}$ is in $\mathcal{R}(C_{XX}^{\beta})$ with $\beta \geq 0$, and $E[f(Z)|W = \cdot] \in \mathcal{R}(C_{WW}^{\nu})$ for some $\nu \geq 0$. For the regularization constants $\varepsilon_n = n^{-\max\{\frac{2}{3}\alpha, \frac{1}{1+\beta}\alpha\}}$ and $\delta_n = n^{-\max\{\frac{4}{9}\gamma, \frac{4}{2\nu+5}\gamma\}}$, where $\gamma = \min\{\frac{2}{3}\alpha, \frac{2\beta+1}{2\beta+2}\alpha\}$, we have for any $y \in \mathcal{Y}$*

$$\mathbf{f}_X^T R_{X|Y} \mathbf{k}_Y(y) - E[f(Z)|W = y] = O_p(n^{-\min\{\frac{4}{9}\gamma, \frac{2\nu}{2\nu+5}\gamma\}}), \quad (n \to \infty),$$

where $\mathbf{f}_X^T R_{X|Y} \mathbf{k}_Y(y)$ is given by Equation (13).

Theorem 15 *Let f be a function in $\mathcal{H}_{\mathcal{X}}$, (Z,W) be a random variable that has the distribution Q defined by Equation (5), and $\widehat{m}_{\Pi}^{(n)}$ be an estimator of m_{Π} such that $\|\widehat{m}_{\Pi}^{(n)} - m_{\Pi}\|_{\mathcal{H}_{\mathcal{X}}} = O_p(n^{-\alpha})$ $(n \to \infty)$ for some $0 < \alpha \leq 1/2$. Assume that the Radon Nikodym derivative $d\Pi/dP_{\mathcal{X}}$ is in $\mathcal{R}(C_{XX}^{\beta})$ with $\beta \geq 0$, $E[f(Z)|W = \cdot] \in \mathcal{H}_{\mathcal{Y}}$, and $J_{\mathcal{Y}} E[f(Z)|W = \cdot] \in \mathcal{R}(\tilde{C}_{WW}^{\nu})$ for some $\nu > 0$. With the regularization constants $\varepsilon_n = n^{-\max\{\frac{2}{3}\alpha, \frac{1}{1+\beta}\alpha\}}$ and $\delta_n = n^{-\max\{\frac{1}{2}\gamma, \frac{2}{\nu+2}\gamma\}}$, where $\gamma = \min\{\frac{2}{3}\alpha, \frac{2\beta+1}{2\beta+2}\alpha\}$, we have*

$$\left\| \mathbf{f}_X^T R_{X|Y} \mathbf{k}_Y(W) - E[f(Z)|W] \right\|_{L^2(Q_{\mathcal{Y}})} = O_p(n^{-\min\{\frac{1}{2}\gamma, \frac{\nu}{\nu+2}\gamma\}}),$$

as n goes to infinity.

Acknowledgments

We would like to express our gratitude to Action Editor and anonymous referees for their helpful feedback and suggestions. We also thank Arnaud Doucet, Lorenzo Rosasco, Yee Whye Teh and Shuhei Mano for their valuable comments. KF has been supported in part by JSPS KAKENHI (B) 22300098 and MEXT KAKENHI on Innovative Areas 25120012. LS is supported in part by NSF IIS1218749 and NIH BIGDATA 1R01GM108341-01. AG has been supported in part by the MPI for Intelligent Systems.

References

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

Francis R. Bach and Michael I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

Charles R. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

Alain Berlinet and Christine Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publisher, 2004.

David Blei and Michael Jordan. Variational inference for dirichlet process mixtures. *Journal of Bayesian Analysis*, 1(1):121–144, 2006.

Byron Boots, Arthur Gretton, and Geoffrey J. Gordon. Hilbert space embeddings of predictive state representations. In *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI2013)*, pages 92–101, 2013.

Aedian W. Bowman. An alternative method of cross-validation for the smoothing of density estimates. *Biometrika*, 71(2):353–360, 1984.

Andrea Caponnetto and Ernesto De Vito. Optimal rates for regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

Arnaud Doucet, Nando De Freitas, and Neil J. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

Mona Eberts and Ingo Steinwart. Optimal learning rates for least squares svms using gaussian kernels. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1539–1547. Curran Associates, Inc., 2011.

Heinz W. Engl, Martin Hanke, and Andreas Neubauer. *Regularization of Inverse Problems*. Kluwer Academic Publishers, 2000.

Shai Fine and Katya Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

Kenji Fukumizu, Francis R. Bach, and Michael I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

Kenji Fukumizu, Arthur Gretton, Xiaohai Sun, and Bernhard Schölkopf. Kernel measures of conditional dependence. In *Advances in Neural Information Processing Systems 20*, pages 489–496. MIT Press, 2008.

Kenji Fukumizu, Francis R. Bach, and Michael I. Jordan. Kernel dimension reduction in regression. *Annals of Statistics*, 37(4):1871–1905, 2009a.

Kenji Fukumizu, Bhrath Sriperumbudur, Arthur Gretton, and Bernhard Schölkopf. Characteristic kernels on groups and semigroups. In *Advances in Neural Information Processing Systems 21*, pages 473–480, Red Hook, NY, 2009b. Curran Associates Inc.

Arthur Gretton, Karsten M. Borgwardt, Malte Rasch, Bernhard 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, Cambridge, MA, 2007. MIT Press.

Arthur Gretton, Kenji Fukumizu, Choon Hui Teo, Le Song, Bernhard Schölkopf, and Alex Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20*, pages 585–592. MIT Press, 2008.

Arthur Gretton, Kenji Fukumizu, Zaid Harchaoui, and Bharath Sriperumbudur. A fast, consistent kernel two-sample test. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 673–681. 2009a.

Arthur Gretton, Kenji Fukumizu, and Bharath K. Sriperumbudur. Discussion of: Brownian distance covariance. *Annals of Applied Statistics*, 3(4):1285–1294, 2009b.

Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

Steffan Grünewälder, Guy Lever, Luca Baldassarre, Sam Patterson, Arthur Gretton, and Massimiliano Pontil. Conditional mean embeddings as regressors. In *Proceedings of the 29th International Conference on Machine Learning (ICML2012)*, pages 1823–1830, 2012.

Steffen Grünewälder, Arthur Gretton, and John Shawe-Taylor. Smooth operators. In *Proceedings of the 30th International Conference on Machine Learning (ICML2013)*, pages 1184–1192, 2013.

Zaid Harchaoui, Francis Bach, and Eric Moulines. Testing for homogeneity with kernel Fisher discriminant analysis. In *Advances in Neural Information Processing Systems 21*, pages 609–616, Cambridge, MA, 2008. MIT Press.

Simon J. Julier and Jeffrey K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defence Sensing, Simulation and Controls*, 1997.

Annaliisa Kankainen and Nikolai G. Ushakov. A consistent modification of a test for independence based on the empirical characteristic function. *Journal of Mathematical Sciencies*, 89:1582–1589, 1998.

Steven MacEachern. Estimating normal means with a conjugate style Dirichlet process prior. *Communications in Statistics – Simulation and Computation*, 23(3):727–741, 1994.

Steven N. MacEachern, Merlise Clyde, and Jun S. Liu. Sequential importance sampling for nonparametric Bayes models: The next generation. *The Canadian Journal of Statistics*, 27(2):251–267, 1999.

Paul Marjoram, John Molitor, Vincent Plagnol, and Simon Tavare. Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003.

Florence Merlevéde, Magda Peligrad, and Sergey Utev. Sharp conditions for the clt of linear processes in a hilbert space. *Journal of Theoretical Probability*, 10:681–693, 1997.

Charles A. Micchelli and Massimiliano Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, 2005.

Sebastian Mika, Bernhard Schölkopf, Alex Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel PCA and de-noising in feature spaces. In *Advances in Neural Information Pecessing Systems 11*, pages 536–542. MIT Press, 1999.

Valérie Monbet, Pierre Ailliot, and Pierre-François Marteau. l^1-convergence of smoothing densities in non-parametric state space models. *Statistical Inference for Stochastic Processes*, 11:311–325, 2008.

Peter Müller and Fernando A. Quintana. Nonparametric Bayesian data analysis. *Statistical Science*, 19(1):95–110, 2004.

Shigeki Nakagome, Kenji Fukumizu, and Shuhei Mano. Kernel approximate Bayesian computation for population genetic inferences. *Statistical Applications in Genetics and Molecular Biology*, 2013. Accepted.

Yu Nishiyama, Abdeslam Boularias, Arthur Gretton, and Kenji Fukumizu. Hilbert space embeddings of POMDPs. In *Proceedigns of Conference on Uncertainty in Artificial Intelligence (UAI2012)*, pages 644–653, 2012.

Mats Rudemo. Empirical choice of histograms and kernel density estimators. *Scandinavian Journal of Statistics*, 9(2):pp. 65–78, 1982.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

Albert N. Shiryaev. *Probability*. Springer, 2nd edition, 1995.

Scott A. Sisson, Yanan Fan, and Mark M. Tanaka. Sequential Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6):1760–1765, 2007.

Steve Smale and Ding-Xuan Zhou. Learning theory estimates via integral operators and their approximation. *Constructive Approximation*, 26:153–172, 2007.

Le Song, Jonathan Huang, Alexander Smola, and Kenji Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th International Conference on Machine Learning (ICML2009)*, pages 961–968, 2009.

Le Song, Arthur Gretton., and Carlos Guestrin. Nonparametric tree graphical models via kernel embeddings. In *Proceedings of AISTATS 2010*, pages 765–772, 2010a.

Le Song, Sajid M. Siddiqi, Geoffrey Gordon, and Alexander Smola. Hilbert space embeddings of hidden Markov models. In *Proceedings of the 27th International Conference on Machine Learning (ICML2010)*, pages 991–998, 2010b.

Le Song, Arthur Gretton, Danny Bickson, Yucheng Low, and Carlos Guestrin. Kernel belief propagation. In *Proceedings of AISTATS 2011*, pages 707–715, 2011.

Bharath K. Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert R.G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010.

Bharath K. Sriperumbudur, Kenji Fukumizu, and Gert Lanckriet. Universality, characteristic kernels and RKHS embedding of measures. *Journal of Machine Learning Research*, 12:2389–2410, 2011.

John Stachurski. A Hilbert space central limit theorem for geometrically ergodic Markov chains. Working paper, Australian National University, 2012.

Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer, 2008.

Charles J. Stone. Optimal global rates of convergence for nonparametric regression. *Annals of Statistics*, 10(4):1040–1053, 1982.

Simon Tavaré, David J. Balding, Robert C. Griffithis, and Peter Donnelly. Inferring coalescence times from dna sequece data. *Genetics*, 145:505–518, 1997.

Sebastian Thrun, John Langford, and Dieter Fox. Monte Carlo hidden Markov models: Learning non-parametric models of partially observable stochastic processes. In *Proceedings of International Conference on Machine Learning (ICML 1999)*, pages 415–424, 1999.

Larry Wasserman. *All of Nonparametric Statistics*. Springer, 2006.

Mike West, Peter Müller, and Michael D. Escobar. Hierarchical priors and mixture models, with applications in regression and density estimation. In P. Freeman et al, editor, *Aspects of Uncertainty: A Tribute to D.V. Lindley*, pages 363–386. Wiley, 1994.

Harold Widom. Asymptotic behavior of the eigenvalues of certain integral equations. *Transactions of the American Mathematical Society*, 109:278–295, 1963.

Harold Widom. Asymptotic behavior of the eigenvalues of certain integral equations II. *Archive for Rational Mechanics and Analysis*, 17:215–229, 1964.

Kun Zhang, Jan Peters, Dominik Janzing, and Bernhard Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *27th Conference on Uncertainty in Artificial Intelligence*, pages 804–813, 2011.

Journal of Machine Learning Research 14 (2013) 3785-3812 Submitted 11/12; Revised 8/13; Published 12/13

Optimally Fuzzy Temporal Memory

Karthik H. Shankar SHANKARK@BU.EDU
Marc W. Howard MARC777@BU.EDU
Center for Memory and Brain
Boston University
Boston, MA 02215, USA

Editor: Peter Dayan

Abstract

Any learner with the ability to predict the future of a structured time-varying signal must maintain a memory of the recent past. If the signal has a characteristic timescale relevant to future prediction, the memory can be a simple shift register—a moving window extending into the past, requiring storage resources that linearly grows with the timescale to be represented. However, an independent general purpose learner cannot *a priori* know the characteristic prediction-relevant timescale of the signal. Moreover, many naturally occurring signals show scale-free long range correlations implying that the natural prediction-relevant timescale is essentially unbounded. Hence the learner should maintain information from the longest possible timescale allowed by resource availability. Here we construct a fuzzy memory system that optimally sacrifices the temporal accuracy of information in a scale-free fashion in order to represent prediction-relevant information from exponentially long timescales. Using several illustrative examples, we demonstrate the advantage of the fuzzy memory system over a shift register in time series forecasting of natural signals. When the available storage resources are limited, we suggest that a general purpose learner would be better off committing to such a fuzzy memory system.

Keywords: temporal information compression, forecasting long range correlated time series

1. Introduction

Natural learners face a severe computational problem in attempting to predict the future of time varying signals. Rather than being presented with a large training set of examples, they must compute on-line using a continuously evolving representation of the recent past. A basic question arises here—how much of the recent past is required to generate future predictions? Maintaining past information in memory comes with a metabolic cost; we would expect a strong evolutionary pressure to minimize the resources required. A shift register can accurately represent information from the recent past up to a chosen timescale, while consuming resources that grow linearly with that timescale. However, the prediction-relevant timescale of the signal is generally unknown prior to learning. Moreover there are many examples of naturally occurring signals with scale-free long range correlations (Voss and Clarke, 1975; Mandelbrot, 1982; Field, 1987; Torralba and Oliva, 2003; Linkenkaer-Hansen et al., 2001; Baillie, 1996; Gilden, 2001; Van Orden et al., 2003; Wagenmakers et al., 2004), commonly known as $1/f$ signals, making the natural prediction-relevant timescale essentially unbounded. Our focus is on the following question: If an independent general purpose learner is to forecast long range correlated natural signals, what is the optimal way to represent the past information in memory with limited resources?

We argue that the solution is to construct a memory that reflects the natural scale-free temporal structure associated with the uncertainties of the world. For example, the timing of an event that happened 100 seconds ago does not have to be represented as accurately in memory as the timing of an event that happened 10 seconds ago. Sacrificing the temporal accuracy of information in memory leads to tremendous resource conservation, yielding the capacity to represent information from exponentially long timescales with linearly growing resources. Moreover, by sacrificing the temporal accuracy of information in a scale-free fashion, the learner can gather the relevant statistics from the signal in a way that is optimal if the signal contains scale-free fluctuations. To mechanistically construct such a memory system, it is imperative to keep in mind that the information represented in memory should *self sufficiently* evolve in real time without relying on any information other than the instantaneous input and what is already represented in memory; reliance on any external information would require additional storage resources. In this paper we describe a *Fuzzy* (meaning temporally inaccurate) memory system that (i) represents information from very long timescales under limited resources, (ii) optimally sacrifices temporal accuracy while maximally preserving the prediction-relevant information from the past, and (iii) evolves self sufficiently in real time.[1]

The layout of the paper is as follows. In Section 2, based on some general properties of long range correlated signals, we derive the criterion for optimally sacrificing the temporal accuracy so that the prediction relevant information from exponentially long time scales is maximally preserved in the memory with finite resources. However, it is non-trivial to construct such a memory in a self sufficient way. In Section 3, we describe a strategy to construct a self sufficient scale-free representation of the recent past. This strategy is based on a neuro-cognitive model of internal time, TILT (Shankar and Howard, 2012), and is mathematically equivalent to encoding the Laplace transform of the past and approximating its inverse to reconstruct a fuzzy representation of the past. With an optimal choice of a set of memory nodes, this representation naturally leads to a self-sufficient fuzzy memory system. In Section 4, we illustrate the utility of the fuzzy memory with some simple time series forecasting examples. We show that the fuzzy memory enhances the ability to predict the future in comparison to a shift register with equal number of nodes. Optimal representation of the recent past in memory does not by itself guarantee the ability to successfully predict the future, for it is crucial to learn the prediction-relevant statistics underlying the signal with an efficient learning algorithm. The choice of the learning algorithm is however largely modular to the choice of the memory system. Here we entirely sidestep the problem of learning, and only focus on the memory. As a place holder for a learning algorithm, we use linear regression in the demonstrations of time series forecasting.

2. Optimal Accuracy-Capacity Tradeoff

Suppose a learner needs to learn to predict a real valued time series with long range correlations. Let $V = \{v_n : n \in (1,2,3...\infty)\}$ represent all past values of the time series relative to the present moment at $n = 0$. Let V be a stationary series with zero mean and finite variance. Ignoring any higher order correlations, let the two point correlation be $\langle v_n v_m \rangle \simeq 1/|n-m|^{\alpha}$. When $\alpha \leq 1$, the series will be long range correlated (Beran, 1994). The goal of the learner is to successfully predict the current value v_o at $n = 0$. Figure 1 shows a sample time series leading up to the present moment. The y-axis corresponds to the present moment and to its right lies the unknown future values of the time series. The x-axis labeled as shift register denotes a memory buffer wherein each v_n is accurately stored in

1. Fuzzy temporal memory is not related to fuzzy logic.

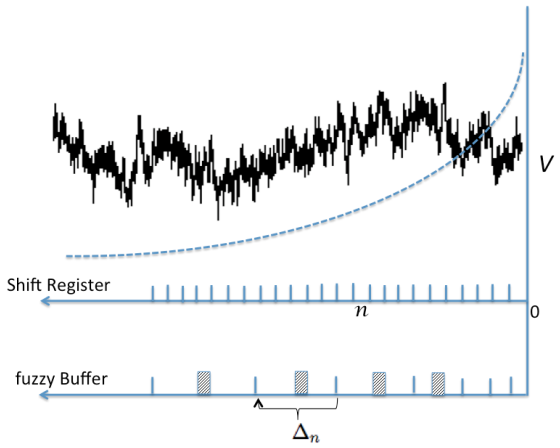


Figure 1: A sample time series V with power-law two-point correlation is plotted w.r.t. time(n) with the current time step taken to be $n = 0$. The figure contrasts the way in which the information is represented in a shift register and the fuzzy buffer. Each node of the fuzzy buffer linearly combines information from a bin containing multiple shift register nodes. The dashed curve shows the predictive information content from each time step that is relevant for predicting the future value of the time series.

a unique node. As we step forward in time, the information stored in each node will be transferred to its left-neighbor and the current value v_o will enter the first node. The shift register can thus self sufficiently evolve in real time. The longest time scale that can be represented with the shift register is linearly related to the number of nodes.

Given a limited number of memory nodes, what is the optimal way to represent V in the memory so that the information relevant to prediction of v_o is maximally preserved? We will show that this is achieved in the fuzzy buffer shown in Figure 1. Each node of the fuzzy buffer holds the average of v_ns over a bin. For the fuzzy buffer, the widths of the bins increase linearly with the center of the bin; the bins are chosen to tile the past time line. Clearly, the accuracy of representing V is sacrificed in the process of compressing the information in an entire bin into a real number, but note that we attain the capacity to represent information from exponentially long timescales. With some analysis, we will show that sacrificing the accuracy with bin widths chosen in this way leads to maximal preservation of prediction-relevant information.

It is clear that the fuzzy buffer shown in Figure 1 cannot evolve self sufficiently in real time; information lost during compression of a bin at any moment is required at the next moment to recompute the bin average. At each moment we would explicitly require all the v_ns to correctly update the values in the fuzzy buffer. Even though such a fuzzy buffer is not self sufficient, we shall analyze it to derive the optimal binning strategy that maximally preserves prediction-relevant information from the past . The reader who is willing to take the optimality of linearly-increasing bin widths on faith can skip ahead to Section 3 where we construct a self-sufficient memory system with that property.

2.1 Deriving the Optimal Binning Strategy

To quantify the prediction relevant information contained in V, let us first review some general properties of the long range correlated series. Since our aim here is only to represent V in memory and not to actually understand the generating mechanism underlying V, it is sufficient to consider V as being generated by a generic statistical algorithm, the ARFIMA model (Granger and Joyeux, 1980; Hosking, 1981; Wagenmakers et al., 2004). The basic idea behind the ARFIMA algorithm is that white noise at each time step can be fractionally integrated to generate a time series with long range correlations.[2] Hence the time series can be viewed as being generated by an infinite auto-regressive generating function. In other words, v_o can be generated from the past series V and an instantaneous Gaussian white noise input η_o.

$$v_o = \eta_o + \sum_{n=1}^{\infty} a(n) v_n. \tag{1}$$

The ARFIMA algorithm specifies the coefficients $a(n)$ in terms of the exponent α in the two point correlation function.

$$a(n) = \frac{(-1)^{n+1}\Gamma(d+1)}{\Gamma(n+1)\Gamma(d-n+1)}, \tag{2}$$

where d is the fractional integration power given by $d = (1-\alpha)/2$. The time series is stationary and long range correlated with finite variance only when $d \in (0, 1/2)$ or $\alpha \in (0,1)$ (Granger and Joyeux, 1980; Hosking, 1981). The asymptotic behavior of $a(n)$ for large n can be obtained by applying Euler's reflection formula and Stirling's formula to approximate the Gamma functions in Equation 2. It turns out that when either d is small or n is large,

$$a(n) \simeq \left[\frac{\Gamma(d+1)\sin(\pi d)}{\pi}\right] n^{-(1+d)}. \tag{3}$$

For the sake of analytic tractability, the following analysis will focus only on small values of d. From Equation 1, note that $a(n)$ is a measure of the relevance of v_n in predicting v_o, which we shall call the $\mathcal{PIC}$-Predictive Information Content of v_n. Taylor expansion of Equation 3 shows that each $a(n)$ is linear in d up to the leading order. Hence in small d limit, any v_n is a stochastic term η_n plus a history dependent term of order $O(d)$. Restricting to linear order in d, the total $\mathcal{PIC}$ of v_n and v_m is simply the sum of their individual $\mathcal{PIC}$s, namely $a(n)+a(m)$. Thus in the small d limit, $\mathcal{PIC}$ is a simple measure of predictive information that is also an extensive quantity.

When the entire V is accurately represented in an infinite shift register, the total $\mathcal{PIC}$ contained in the shift register is the sum of all $a(n)$. This is clearly the maximum attainable value of $\mathcal{PIC}$ in any memory buffer, and it turns out to be 1.

$$\mathcal{PIC}_{max} = \sum_{n=1}^{\infty} a(n) = 1.$$

In a shift register with $N_{\max}$ nodes, the total $\mathcal{PIC}$ is

$$\begin{aligned} \mathcal{PIC}^{\mathrm{SR}}_{\mathrm{tot}} = \sum_{n=1}^{N_{\max}} a(n) \quad &= \quad 1 - \sum_{N_{\max}}^{\infty} a(n) \;\simeq\; 1 - \frac{\sin(\pi d)\Gamma(d+1)}{\pi d} N_{\max}^{-d}, \\ &\xrightarrow{d\to 0} \quad d \ln N_{\max}. \end{aligned} \tag{4}$$

2. The most general model ARFIMA(p,d,q) can generate time series with both long and short range structures. Here we choose $p = q = 0$ to ignore the short range structures.

For any fixed $N_{\max}$, when d is sufficiently small $\mathcal{PIC}_{\rm tot}^{\rm SR}$ will be very small. For example, when $N_{\max} = 100$ and $d = 0.1$, $\mathcal{PIC}_{\rm tot}^{\rm SR} \simeq 0.4$. For smaller values of d, observed in many natural signals, $\mathcal{PIC}_{\rm tot}^{\rm SR}$ would be even lower. When $d = 0.01$, for $N_{\max}$ as large as 10000, the $\mathcal{PIC}_{\rm tot}^{\rm SR}$ is only 0.08. A large portion of the predictive information lies in long timescales. So the shift register is ill-suited to represent information from a long range correlated series.

The $\mathcal{PIC}_{\rm tot}$ for a memory buffer can be increased if each of its nodes stored a linear combination of many v_ns rather than a single v_n as in the shift register. This can be substantiated through information theoretic considerations formulated by the Information Bottleneck ($I\mathcal{B}$) method (Tishby et al., 1999). A multi-dimensional Gaussian variable can be systematically compressed to lower dimensions by linear transformations while maximizing the relevant information content (Chechik et al., 2005). Although V is a unidimensional time series in our consideration, at any moment the entire V can be considered as an infinite dimensional Gaussian variable since only its two point correlations are irreducible. Hence it heuristically follows from $I\mathcal{B}$ that linearly combining the various v_ns into a given number of combinations and representing each of them in separate memory nodes should maximize the $\mathcal{PIC}_{\rm tot}$. By examining Equation 1, it is immediately obvious that if we knew the values of $a(n)$, the entire V could be linearly compressed into a single real number $\sum a(n)v_n$ conveying all of the prediction relevant information. However, such a single-node memory buffer is not self sufficient: at each moment we explicitly need the entire V to update the value in that node. As an unbiased choice that does not require *a priori* knowledge of the statistics of the time series, we simply consider uniform averaging over a bin. Uniform averaging over a bin discards separate information about the time of the values contributing to the bin. Given this consideration, how should we space the bins to maximize the $\mathcal{PIC}_{\rm tot}$?

Consider a bin ranging between n and $n+\Delta_n$. We shall examine the effect of averaging all the v_ms within this bin and representing it in a single memory node. If all the v_ms in the bin are individually represented, then the $\mathcal{PIC}$ of the bin is $\sum_m a(m)$. Compressing all the v_ms into their average would however lead to an error in prediction of v_o; from Equation 1 this error is directly related to the extent to which the $a(m)$s within the bin are different from each other. Hence there should be a reduction in the $\mathcal{PIC}$ of the bin. Given the monotonic functional form of $a(n)$, the maximum reduction can only be $\sum_m |a(m)-a(n)|$. The net $\mathcal{PIC}$ of the memory node representing the bin average is then

$$\begin{aligned}\mathcal{PIC} &= \sum_{m=n}^{n+\Delta_n} a(m) - \sum_{m=n}^{n+\Delta_n} |a(n)-a(m)|, \\ &\simeq \frac{n\,a(n)}{d}\left[2-2\left(1+\frac{\Delta_n}{n}\right)^{-d}-\frac{d\,\Delta_n}{n}\right].\end{aligned}$$

The series summation in the above equation is performed by approximating it as an integral in the large n limit. The bin size that maximizes the $\mathcal{PIC}$ can be computed by setting the derivative of $\mathcal{PIC}$ w.r.t. Δ_n equal to zero. The optimal bin size and the corresponding $\mathcal{PIC}$ turns out to be

$$\Delta_n^{opt} = \left[2^{1/(1+d)}-1\right] n, \qquad \mathcal{PIC}^{opt} \simeq \frac{n\,a(n)}{d}\left[2+d-(1+d)2^{1/(1+d)}\right]. \tag{5}$$

When the total number of nodes $N_{\max}$ is finite, and we want to represent information from the longest possible timescale, the straightforward choice is to pick successive bins such that they completely tile up the past time line as schematically shown by fuzzy buffer in Figure 1. If we label

the nodes of the fuzzy buffer by N, ranging from 1 to $N_{\max}$, and denote the starting point of each bin by n_N, then

$$n_{N+1} = (1+c)n_N \qquad \Longrightarrow \qquad n_N = n_1(1+c)^{(N-1)}, \tag{6}$$

where $1+c = 2^{1/(1+d)}$. Note that the fuzzy buffer can represent information from timescales of the order $n_{N_{\max}}$, which is exponentially large compared to the timescales represented by a shift register with $N_{\max}$ nodes. The total $\mathcal{PIC}$ of the fuzzy buffer, $\mathcal{PIC}^{\text{FB}}_{\text{tot}}$, can now be calculated by summing over the $\mathcal{PIC}$s of each of the bins. Focusing on small values of d so that $a(n)$ has the power law form for all n, applying Equations 3 and 5 yields

$$\mathcal{PIC}^{\text{FB}}_{\text{tot}} \simeq \sum_{N=1}^{N_{\max}} \frac{\sin(\pi d)\Gamma(d+1)}{\pi d} \left[2+d-(1+d)2^{1/(1+d)}\right] n_N^{-d}.$$

Taking $n_1 = 1$ and n_N given by Equation 6,

$$\begin{aligned} \mathcal{PIC}^{\text{FB}}_{\text{tot}} \quad &\simeq \quad \frac{\sin(\pi d)\Gamma(d+1)}{\pi d} \left[2+d-(1+d)2^{1/(1+d)}\right] \left[\frac{1-(1+c)^{-d\ N_{\max}}}{1-(1+c)^{-d}}\right], \\ &\xrightarrow{d\to 0} \quad [\ln 4 - 1] d\ N_{\max}. \end{aligned} \tag{7}$$

Comparing Equations 4 and 7, note that when d is small, the $\mathcal{PIC}^{\text{FB}}_{\text{tot}}$ of the fuzzy buffer grows linearly with $N_{\max}$ while the $\mathcal{PIC}^{\text{SR}}_{\text{tot}}$ of the shift register grows logarithmically with $N_{\max}$. For example, with $N_{\max} = 100$ and $d = 0.01$, the $\mathcal{PIC}^{\text{FB}}_{\text{tot}}$ of the fuzzy buffer is 0.28, while the $\mathcal{PIC}^{\text{SR}}_{\text{tot}}$ of the shift register is only 0.045. Hence when $N_{\max}$ is relatively small, the fuzzy buffer represents a lot more predictive information than a shift register.

The above description of the fuzzy buffer corresponds to the ideal case wherein the neighboring bins do not overlap and uniform averaging is performed within each bin. Its critical property of linearly increasing bin sizes ensures that the temporal accuracy of information is sacrificed optimally and in a scale-free fashion. However, this ideal fuzzy buffer cannot self sufficiently evolve in real time because at every moment all v_ns are explicitly needed for its construction. In the next section, we present a self sufficient memory system that possesses the critical property of the ideal fuzzy buffer, but differs from it by having overlapping bins and non-uniform weighted averaging within the bins. To the extent the self sufficient fuzzy memory system resembles the ideal fuzzy buffer, we can expect it to be useful in representing long range correlated signals in a resource-limited environment.

3. Constructing Self Sufficient Fuzzy Memory

In this section, we first describe a mathematical basis for representing the recent past in a scale-free fashion based on a neuro-cognitive model of internal time, TILT (Shankar and Howard, 2012). We then describe several critical considerations necessary to implement this representation of recent past into a discrete set of memory nodes. Like the ideal fuzzy buffer described in the Section 2, this memory representation will sacrifice temporal accuracy to represent prediction-relevant information over exponential time scales. But unlike the ideal fuzzy buffer, the resulting memory representation will be self sufficient, without requiring additional resources to construct the representation.

Let $\mathbf{f}(\tau)$ be a real valued function presented over real time τ. Our aim now is to construct a memory that represents the past values of $\mathbf{f}(\tau)$ as activity distributed over a set of nodes with

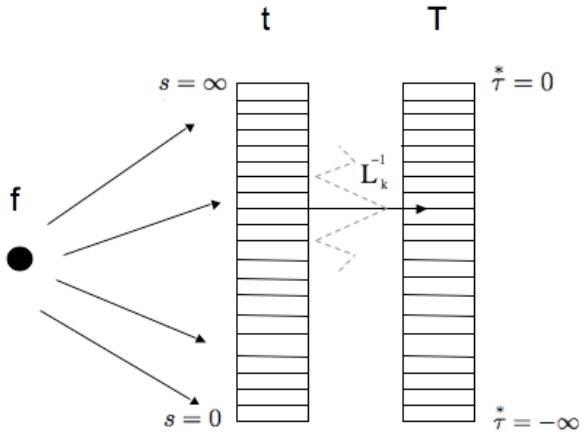


Figure 2: The scale-free fuzzy representation - Each node in the **t** column is a leaky integrator with a specific decay constant s that is driven by the functional value **f** at each moment. The activity of the **t** column is transcribed at each moment by the operator $\mathbf{L}_k^{-1}$ to represent the past functional values in a scale-free fuzzy fashion in the **T** column.

accuracy that falls off in a scale-free fashion. This is achieved using two columns of nodes **t** and **T** as shown in Figure 2. The **T** column estimates $\mathbf{f}(\tau)$ up to the present moment, while the **t** column is an intermediate step used to construct **T**. The nodes in the **t** column are leaky integrators with decay constants denoted by s. Each leaky integrator independently gets activated by the value of **f** at any instant and gradually decays according to

$$\frac{d\mathbf{t}(\tau,s)}{d\tau} = -s\mathbf{t}(\tau,s) + \mathbf{f}(\tau). \tag{8}$$

At every instant, the information in the **t** column is transcribed into the **T** column through a linear operator $\mathbf{L}_k^{-1}$.

$$\begin{aligned} \mathbf{T}(\tau,\overset{*}{\tau}) &= \frac{(-1)^k}{k!} s^{k+1}\mathbf{t}^{(k)}(\tau,s) \ : \ \text{where } s=-k/\overset{*}{\tau}. \\ \mathbf{T} &\equiv \mathbf{L}_k^{-1}\big[\mathbf{t}\big]. \end{aligned} \tag{9}$$

Here k is a positive integer and $\mathbf{t}^{(k)}(\tau,s)$ is the k-th derivative of $\mathbf{t}(\tau,s)$ with respect to s. The nodes of the **T** column are labeled by the parameter $\overset{*}{\tau}$ and are in one to one correspondence with the nodes of the **t** column labeled by s. The correspondence between s and $\overset{*}{\tau}$ is given by $s=-k/\overset{*}{\tau}$. We refer to $\overset{*}{\tau}$ as *internal time*; at any moment τ, a $\overset{*}{\tau}$ node estimates the value of **f** at a time $\tau+\overset{*}{\tau}$ in the past. The range of values of s and $\overset{*}{\tau}$ can be made as large as needed at the cost of resources, but for mathematical idealization we let them have an infinite range.

The mathematical inspiration of this approach comes from the fact that $\mathbf{t}(\tau,s)$ encodes the Laplace transform of the entire history of the function **f** up to time τ, and the operator $\mathbf{L}_k^{-1}$ approximately inverts the Laplace transform (Post, 1930). As $k\to\infty$, $\mathbf{T}(\tau,\overset{*}{\tau})$ becomes a faithful

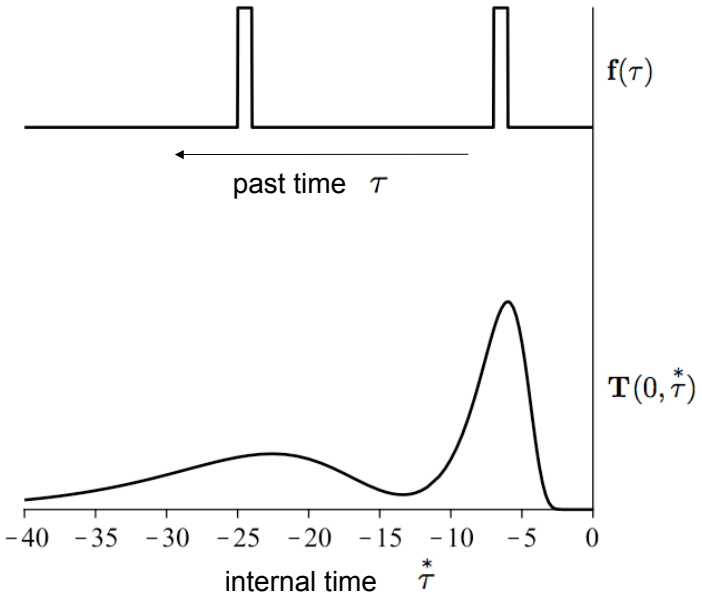


Figure 3: Taking the present moment to be $\tau = 0$, a sample $\mathbf{f}(\tau)$ is plotted in the top, and the momentary activity distributed across the $\mathbf{T}$ column nodes is plotted in the bottom.

reconstruction of the history of $\mathbf{f}$ from $-\infty$ to τ, that is $\mathbf{T}(\tau,\overset{*}{\tau}) \simeq \mathbf{f}(\tau+\overset{*}{\tau})$ for all values of $\overset{*}{\tau}$ from 0 to $-\infty$. When k is finite $\mathbf{T}(\tau,\overset{*}{\tau})$ is an inaccurate reconstruction of the history of $\mathbf{f}$. For example, taking the current moment to be $\tau = 0$, Figure 3 illustrates an $\mathbf{f}$ that is briefly non-zero around $\tau = -7$ and $\tau = -23$. The reconstructed history of $\mathbf{f}$ in the $\mathbf{T}$ column shows two peaks approximately around $\overset{*}{\tau} = -7$ and $\overset{*}{\tau} = -23$. The value of $\mathbf{f}$ at any particular moment in the past is thus smeared over a range of $\overset{*}{\tau}$ values, and this range of smearing increases as we go deeper into the past. Thus, the more distant past is reconstructed with a lesser temporal accuracy.

Furthermore, it turns out that the smear is precisely scale invariant. To illustrate this, consider $\mathbf{f}(\tau)$ to be a Dirac delta function at a moment τ_o in the past, $\mathbf{f}(\tau) = \delta(\tau - \tau_o)$, and let the present moment be $\tau = 0$. Applying Equations 8 and 9, we obtain

$$\mathbf{T}(0,\overset{*}{\tau}) = \frac{1}{|\tau_o|}\frac{k^{k+1}}{k!}\left(\frac{\tau_o}{\overset{*}{\tau}}\right)^{k+1} e^{-k(\tau_o/\overset{*}{\tau})}. \tag{10}$$

In the above equation both τ_o and $\overset{*}{\tau}$ are negative. $\mathbf{T}(0,\overset{*}{\tau})$ is the fuzzy reconstruction of the delta function input. $\mathbf{T}(0,\overset{*}{\tau})$ is a smooth peaked function whose height is proportional to $1/|\tau_o|$, width is proportional to $|\tau_o|$, and the area is equal to 1. Its dependence on the ratio $(\tau_o/\overset{*}{\tau})$ ensures scale invariance—for any τ_o we can linearly scale the $\overset{*}{\tau}$ values to hold the shape of the function fixed. In this sense, $\mathbf{T}$ represents the history of $\mathbf{f}$ with a scale invariant smear. To quantify how much smear is introduced, we can estimate the width of the peak as the standard deviation σ of $\mathbf{T}(0,\overset{*}{\tau})$ from the

above equation, which for $k > 2$ turns out to be

$$\sigma[\mathbf{T}(0,\overset{*}{\tau})] = \frac{|\tau_o|}{\sqrt{k-2}}\left[\frac{k}{k-1}\right]. \tag{11}$$

Note that k is the only free parameter that affects the smear; k indexes the smear in the representation. The larger the k the smaller the smear. In the limit $k \to \infty$, the smear vanishes and the delta function input propagates into the **T** column exactly as delta function without spreading, as if **T** were a shift register.

Though we took **f** to be a simple delta function input to illustrate the scale invariance of the **T** representation, we can easily evaluate the **T** representation of an arbitrary **f** from Equations 8 and 9.

$$\mathbf{T}(0,\overset{*}{\tau}) = \int_{-\infty}^{0}\left[\frac{1}{|\overset{*}{\tau}|}\frac{k^{k+1}}{k!}\left(\frac{\tau'}{\overset{*}{\tau}}\right)^{k} e^{-k(\tau'/\overset{*}{\tau})}\right]\mathbf{f}(\tau')\,d\tau'. \tag{12}$$

The **f** values from a range of past times are linearly combined and represented in each $\overset{*}{\tau}$ node. The term in the square brackets in the above equation is the weighting function of the linear combination. Note that it is not a constant function over a circumscribed past time bin, rather it is a smooth function peaked at $\tau' = \overset{*}{\tau}$, with a spread proportional to $|\overset{*}{\tau}|$. Except for the fact that this weighting function is not uniform, the activity of a $\overset{*}{\tau}$ node has the desired property mimicking a bin of the ideal fuzzy buffer described in Section 2.

3.1 Self Sufficient Discretized Implementation

Although it appears from Equation 12 that the **T** representation requires explicit information about the **f** values over the past time, recall that it can be constructed from instantaneous **t** representation. Since Equation 8 is a local differential equation, the activity of each **t** node will independently evolve in real time, depending only on the present value of the node and the input available at that moment. Hence any discrete set of **t** nodes also evolves self sufficiently in real time. To the extent the activity in a discrete set of **T** nodes can be constructed from a discrete set of **t** nodes, this memory system as a whole can self sufficiently evolve in real time. Since the activity of each $\overset{*}{\tau}$ node in the **T** column is constructed independently of the activity of other $\overset{*}{\tau}$ nodes, we can choose any discrete set of $\overset{*}{\tau}$ values to form our memory system. In accordance with our analysis of the ideal fuzzy buffer in Section 2, we shall pick the following nodes.

$$\overset{*}{\tau}_{min}\,,\ \overset{*}{\tau}_{min}(1+c)\,,\ \overset{*}{\tau}_{min}(1+c)^2\,,\ldots\ \overset{*}{\tau}_{min}(1+c)^{(N_{\max}-1)} = \overset{*}{\tau}_{max}. \tag{13}$$

Together, these nodes form the fuzzy memory representation with $N_{\max}$ nodes. The spacing in Equation 13 will yield several important properties.

Unlike the ideal fuzzy buffer described in Section 2, it is not possible to associate a circumscribed bin for a node because the weighting function (see Equation 12) does not have compact support. Since the weighting function associated with neighboring nodes overlap with each other, it is convenient to view the bins associated with neighboring nodes as partially overlapping. The overlap between neighboring bins implies that some information is redundantly represented in multiple nodes. We shall show in the forthcoming subsections that by appropriately tuning the parameters k and c, this information redundancy can be minimized and equally spread in a scale-free fashion.

3.1.1 DISCRETIZED DERIVATIVE

Although any set of $\overset{*}{\tau}$ nodes could be picked to form the memory buffer, their activity is ultimately constructed from the nodes in the $\mathbf{t}$ column whose s values are given by the one to one correspondence $s = -k/\overset{*}{\tau}$. Since the $\mathbf{L}_k^{-1}$ operator has to take the k-th derivative along the s axis, $\mathbf{L}_k^{-1}$ depends on the way the s-axis is discretized.

For any discrete set of s values, we can define a linear operator that implements a discretized derivative. For notational convenience, let us denote the activity at any moment $\mathbf{t}(\tau, s)$ as simply $\mathbf{t}(s)$. Since $\mathbf{t}$ is a column vector with the rows labeled by s, we can construct a derivative matrix [D] such that

$$\mathbf{t}^{(1)} = [\mathrm{D}]\mathbf{t} \qquad \Longrightarrow \qquad \mathbf{t}^{(k)} = [\mathrm{D}]^k \mathbf{t}.$$

The individual elements in the square matrix [D] depends on the set of s values. To compute these elements, consider any three successive nodes with s values s_{-1}, s_o, s_1. The discretized first derivative of $\mathbf{t}$ at s_o is given by

$$\mathbf{t}^{(1)}(s_o) = \frac{\mathbf{t}(s_1) - \mathbf{t}(s_o)}{s_1 - s_o}\left[\frac{s_o - s_{-1}}{s_1 - s_{-1}}\right] + \frac{\mathbf{t}(s_o) - \mathbf{t}(s_{-1})}{s_o - s_{-1}}\left[\frac{s_1 - s_o}{s_1 - s_{-1}}\right].$$

The row in [D] corresponding to s_o will have non-zero entries only in the columns corresponding to s_{-1}, s_o and s_1. These three entries can be read out as coefficients of $\mathbf{t}(s_{-1})$, $\mathbf{t}(s_o)$ and $\mathbf{t}(s_1)$ respectively in the r.h.s of the above equation. Thus the entire matrix [D] can be constructed from any chosen set of s values. By taking the k-th power of [D], the $\mathbf{L}_k^{-1}$ operator can be straightforwardly constructed and the activity of the chosen set of $\overset{*}{\tau}$ nodes can be calculated at each moment.[3] This memory system can thus self sufficiently evolve in real time.

When the spacing between the nodes (controlled by the parameter c) is small, the discretized k-th derivative will be accurate. Under uniform discretization of the s axis, it can be shown that the relative error in computation of the k-th derivative due to discretization is of the order $O(k\delta_s^2/24)$, where δ_s is the distance between neighboring s values (see appendix B of Shankar and Howard, 2012). Based on the s values corresponding to Equation 13, it turns out that the relative error in the construction of the activity of a $\overset{*}{\tau}$ node is $O(k^3c^2/96\overset{*}{\tau}^2)$. For large $\overset{*}{\tau}$ the error is quite small but for small $\overset{*}{\tau}$ the error can be significant. To curtail the discretization error, we need to hold $\overset{*}{\tau}_{min}$ sufficiently far from zero. The error can also be controlled by choosing small c for large k and vice versa. If for practical purposes we require a very small $\overset{*}{\tau}_{min}$, then ad hoc strategies can be adopted to control the error at low $\overset{*}{\tau}$ nodes. For example, by relaxing the requirement of one to one correspondence between the $\mathbf{t}$ and $\mathbf{T}$ nodes, we can choose a separate set of closely spaced s values to exclusively compute the activity of each of the small $\overset{*}{\tau}$ nodes.

Finally, it has to be noted that the discretization error induced in this process should not be considered as an error in the conventional sense of numerical solutions to differential equations. While numerically evolving differential equations with time-varying boundary conditions, the discretization error in the derivatives will propagate leading to large errors as we move farther away from the boundary at late times. But in the situation at hand, since the activity of each $\overset{*}{\tau}$ node is computed independently of others, the discretization error does not propagate. Moreover, it should be noted

3. Note that we need k extra nodes in the top and bottom of the $\mathbf{t}$ column in addition to those that come from one to one correspondence with the chosen $\overset{*}{\tau}$ values.

that the effect of discretization can be better viewed as coarse-graining the k-th derivative rather than as inducing an error in computing the k-th derivative. The fact that each $\overset{*}{\tau}$ node ultimately holds a scale-free coarse-grained value of the input function (see Equation 12), suggests that we wouldn't need the exact k-th derivative to construct its activity. To the extent the discretization is scale-free as in Equation 13, the $\mathbf{T}$ representation constructed from the coarse-grained k-th derivative will represent some scale free coarse grained value of the input; however the weighting function would not exactly match that in Equation 12. In other words, even if the discretized implementation does not accurately match the continuum limit, it still accurately satisfies the basic properties we require from a fuzzy memory system.

3.1.2 SIGNAL-TO-NOISE RATIO WITH OPTIMALLY-SPACED NODES

The linearity of Equations 8 and 9 implies that any noise in the input function $\mathbf{f}(\tau)$ will exactly be represented in $\mathbf{T}$ without any amplification. However when there is random uncorrelated noise in the $\mathbf{t}$ nodes, the discretized $\mathbf{L}_k^{-1}$ operator can amplify that noise, more so if c is very small. It turns out that choice of nodes according to Equation 13 results in a constant signal-to-noise ratio across time scales.

If uncorrelated noise with standard deviation η is added to the activation of each of the $\mathbf{t}$ nodes, then the $\mathbf{L}_k^{-1}$ operator combines the noise from $2k$ neighboring nodes leading to a noisy $\mathbf{T}$ representation. If the spacing between the nodes neighboring a s node is δ_s, then the standard deviation of the noise generated by the $\mathbf{L}_k^{-1}$ operator is approximately $\eta\sqrt{2^k}s^{k+1}/\delta_s^k k!$. To view this noise in an appropriate context, we can compare it to the magnitude of the representation of a delta function signal at a past time $\tau_o = -k/s$ (see Equation 10). The magnitude of the $\mathbf{T}$ representation for a delta function signal is approximately $k^k e^{-k} s/k!$. The signal to noise ratio (SNR) for a delta function signal is then

$$\text{SNR} = \eta^{-1}\left(\frac{k[\delta_s/s]}{\sqrt{2}e}\right)^k. \tag{14}$$

If the spacing between neighboring nodes changes such that $[\delta_s/s]$ remains a constant for all s, then the signal to noise ratio will remain constant over all timescales. This would however require that the nodes are picked according to Equation 13, making SNR = $\eta^{-1}(kc/\sqrt{2}e)^k$. Any other set of nodes would make the signal to noise ratio zero either at large or small timescales.

This calculation however does not represent the most realistic situation. Because the $\mathbf{t}$ nodes are leaky integrators (Equation 8), the white noise present across time will accumulate and hence nodes with long time constants should have a higher value of η. In fact, the standard deviation of white noise in the $\mathbf{t}$ nodes should go down with s according to $\eta \propto 1/\sqrt{s}$. From Equation 14, we can then conclude that the SNR of large $\overset{*}{\tau}$ nodes should drop down to zero as $1/\sqrt{|\overset{*}{\tau}|}$. However, because each $\overset{*}{\tau}$ node represents a weighted temporal average of the past signal, it is not appropriate to use an isolated delta function signal to estimate the signal to noise ratio. It is more appropriate to compare temporally averaged noise to a temporally spread signal. We consider two such signals. (i) Suppose $\mathbf{f}(\tau)$ itself is a temporally uncorrelated white noise like signal. The standard deviation in the activity of a $\overset{*}{\tau}$ node in response to this signal is proportional to $1/\sqrt{|\overset{*}{\tau}|}$ (see Equation 16 in the appendix). The SNR for this temporally-extended signal is a constant over all $\overset{*}{\tau}$ nodes. (ii) Consider a purely positive signal where $\mathbf{f}(\tau)$ is a sequence of delta function spikes generated by a Poisson process.

The total expected number of spikes that would be generated in the timescale of integration of a $\overset{*}{\tau}$ node is simply $\sqrt{|\overset{*}{\tau}|}$. Consequently, the expectation value of the activity of a $\overset{*}{\tau}$ node in response to this signal would be $\sqrt{|\overset{*}{\tau}|}$ multiplied by the magnitude of representation of a single delta function. The SNR for this signal is again a constant over all $\overset{*}{\tau}$ nodes. So, we conclude that for any realistic stationary signal spread out in time, the SNR will be a constant over all timescales as long as the nodes are chosen according to Equation 13.

3.2 Information Redundancy

The fact that a delta function input in $\mathbf{f}$ is smeared over many $\overset{*}{\tau}$ nodes implies that there is a redundancy in information representation in the $\mathbf{T}$ column. In a truly scale-free memory buffer, the redundancy in information representation should be equally spread over all time scales represented in the buffer. The information redundancy can be quantified in terms of the mutual information shared between neighboring nodes in the buffer. In the appendix, it is shown that in the presence of scale free input signals, the mutual information shared by any two neighboring buffer nodes can be a constant only if the $\overset{*}{\tau}$ nodes are distributed according to Equation 13. Consequently information redundancy is uniformly spread only when the $\overset{*}{\tau}$ nodes are given by Equation 13.

The uniform spread of information redundancy can be intuitively understood by analyzing how a delta function input spreads through the buffer nodes as time progresses. Figure 4 shows the activity of the buffer nodes at three points in time following the input. In Figure 4a where the $\overset{*}{\tau}$ values of the buffer nodes are chosen to be equidistant, the activity is smeared over more and more number of nodes as time progresses. This implies that the information redundancy is large in the nodes representing long timescales. In Figure 4b where the $\overset{*}{\tau}$ values of the buffer nodes are chosen according to Equation 13, the activity pattern does not smear, instead the activity as a whole gets translated with an overall reduction in size. The translational invariance of the activity pattern as it passes through the buffer nodes explains why the information redundancy between any two neighboring nodes is a constant. The translational invariance of the activity pattern can be analytically established as follows.

Consider two different values of τ_o in Equation 10, say τ_1 and τ_2. Let the corresponding $\mathbf{T}$ activities be $\mathbf{T}_1(0,\overset{*}{\tau})$ and $\mathbf{T}_2(0,\overset{*}{\tau})$ respectively. If the $\overset{*}{\tau}$ value of the N-th node in the buffer is given by $\overset{*}{\tau}_N$, then the pattern of activity across the nodes is translationally invariant if and only if $\mathbf{T}_1(0,\overset{*}{\tau}_N) \propto \mathbf{T}_2(0,\overset{*}{\tau}_{N+m})$ for some constant integer m. For this to hold true, we need the quantity

$$\frac{\mathbf{T}_1(0,\overset{*}{\tau}_N)}{\mathbf{T}_2(0,\overset{*}{\tau}_{N+m})} = \left(\frac{\tau_1}{\tau_2}\right)^k \left[\frac{\overset{*}{\tau}_{N+m}}{\overset{*}{\tau}_N}\right]^{k+1} e^{k\left[\frac{\tau_1}{\overset{*}{\tau}_N} - \frac{\tau_2}{\overset{*}{\tau}_{N+m}}\right]}$$

to be independent of N. This is possible only when the quantity inside the power law form and the exponential form are separately independent of N. The power law form can be independent of N only if $\overset{*}{\tau}_N \propto (1+c)^N$, which implies the buffer nodes have $\overset{*}{\tau}$ values given by Equation 13. The exponential form is generally dependent on N except when its argument is zero, which happens whenever $(1+c)^m = \tau_2/\tau_1$ for some integer m. For any given τ_1, there are infinitely many τ_2 values for which the condition holds. Moreover when c is small, the condition will approximately hold for

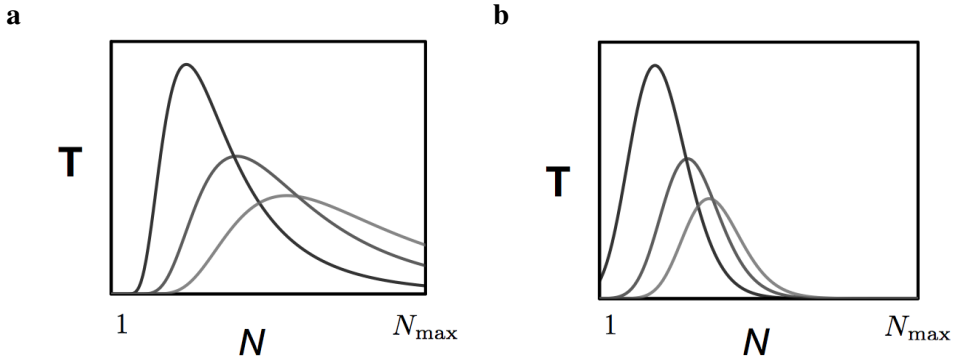


Figure 4: Activity of the fuzzy memory nodes in response to a delta function input at three different times with (**a**) uniformly spaced nodes and (**b**) nodes chosen in accordance with Equation 13.

any τ_2. Hence, the translational invariance of the activity pattern holds only when the $\overset{*}{\tau}$ values of the buffer nodes conform to Equation 13.

3.3 Balancing Information Redundancy and Information Loss

We have seen that the choice of $\overset{*}{\tau}$ nodes in accordance with Equation 13 ensures that the information redundancy is equally distributed over the buffer nodes. However, equal distribution of information redundancy is not sufficient; we would also like to minimize information redundancy. It turns out that we cannot arbitrarily reduce the information redundancy without creating information loss. The parameters k and c have to be tuned in order to balance information redundancy and information loss. If k is too small for a given c, then many nodes in the buffer will respond to input from any given moment in the past, resulting in information redundancy. On the other hand, if k is too large for a given c, the information from many moments in the past will be left unrepresented in any of the buffer nodes, resulting in information loss. So we need to match c with k to simultaneously minimize both information redundancy and information loss.

This can be achieved if the information from any given moment in the past is not distributed over more than two neighboring buffer nodes. To formalize this, consider a delta function input at a time τ_o in the past and let the current moment be $\tau = 0$. Let us look at the activity induced by this input (Equation 10) in four successive buffer nodes, $N-1$, N and $N+1$ and $N+2$. The $\overset{*}{\tau}$ values of these nodes are given by Equation 13, for instance $\overset{*}{\tau}_N = \overset{*}{\tau}_{min}(1+c)^{N-1}$ and $\overset{*}{\tau}_{N+1} = \overset{*}{\tau}_{min}(1+c)^{N}$. From Equation 10, it can be seen that the N-th node attains its maximum activity when $\tau_o = \overset{*}{\tau}_N$ and the $(N+1)$-th node attains its maximum activity when $\tau_o = \overset{*}{\tau}_{N+1}$, and for all the intervening times of τ_o between $\overset{*}{\tau}_N$ and $\overset{*}{\tau}_{N+1}$, the information about the delta function input will be spread over both N-th and the $(N+1)$-th nodes. To minimize the information redundancy, we simply require that when τ_o is in between $\overset{*}{\tau}_N$ and $\overset{*}{\tau}_{N+1}$, all the nodes other than the N-th and the $(N+1)$-th nodes should have almost zero activity.

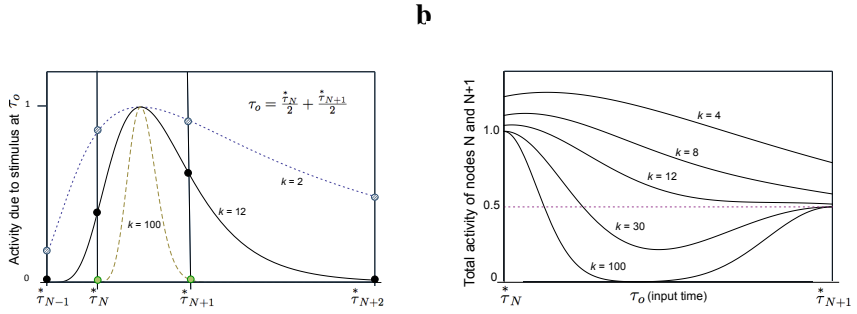


Figure 5: **a**. The activity of four successive fuzzy memory nodes $N-1$, N, $N+1$, and $N+2$ in response to a delta function input at a past moment τ_o that falls right in between the timescales of the N-th and the $(N+1)$-th nodes. The nodes are chosen according to the distribution given by Equation 13 with $c = 1$. **b**. The sum of activity of the N-th and $(N+1)$-th nodes in response to a delta function input at various times τ_o ranging between the timescales of N-th and $(N+1)$-th nodes. For each k, the activities are normalized to have values in the range of 0 to 1.

Figure 5a plots the activity of the four successive nodes with $c = 1$, when τ_o is exactly in the middle of $\overset{*}{\tau}_N$ and $\overset{*}{\tau}_{N+1}$. For each value of k, the activity is normalized so that it lies between 0 and 1. The four vertical lines represent the 4 nodes and the dots represent the activity of the corresponding nodes. Note that for $k = 2$ the activity of all 4 nodes is substantially different from zero, implying a significant information redundancy. At the other extreme, $k = 100$, the activity of all the nodes are almost zero, implying that the information about the delta function input at time $\tau_o = (\overset{*}{\tau}_N + \overset{*}{\tau}_{N+1})/2$ has been lost. To minimize both the information loss and the information redundancy, the value of k should be neither too large nor too small. Note that for $k = 12$, the activities of the $(N-1)$-th and the $(N+2)$-th nodes are almost zero, but activities of the N-th and $(N+1)$-th nodes are non-zero.

For any given c, a rough estimate of the appropriate k can be obtained by matching the difference in the $\overset{*}{\tau}$ values of the neighboring nodes to the smear σ from Equation 11.

$$\sigma = \frac{|\overset{*}{\tau}_{N+1}|}{\sqrt{k-2}} \left[\frac{k}{k-1}\right] \simeq |\overset{*}{\tau}_{N+1} - \overset{*}{\tau}_N| \qquad \Rightarrow \qquad \frac{k}{(k-1)\sqrt{k-2}} \simeq \frac{c}{1+c}. \tag{15}$$

This condition implies that a large value of k will be required when c is small and a small value of k will be required when c is large. In particular, Equation 15 suggests that $k \simeq 8$ when $c = 1$, which will be the parameters we pick for the demonstrations in Section 4.

To further illustrate the information loss at high values of k, Figure 5b shows the sum of activity of the N-th and the $(N+1)$-th nodes for all values of τ_o between $\overset{*}{\tau}_N$ and $\overset{*}{\tau}_{N+1}$. For each k, the activities are normalized so that the N-th node attains 1 when $\tau_o = \overset{*}{\tau}_N$. Focusing on the case of $k = 100$ in Figure 5b, there is a range of τ_o values for which the total activity of the two nodes is very close to zero. The input is represented by the N-th node when τ_o is close to $\overset{*}{\tau}_N$, and is represented by the $(N+1)$-th node when τ_o is close to $\overset{*}{\tau}_{N+1}$, but at intermediate values of τ_o the input is not represented by any node. One way to avoid such information loss is to require that the

total activity of the two nodes not have a local minimum—in other words the minimum should be at the boundary, at $\tau_o = \overset{*}{\tau}_{N+1}$. This is apparent in Figure 5b for k =4, 8 and 12. For $c = 1$, it turns out that there exists a local minimum in the summed activity of the two nodes only for values of k greater than 12. For any given c, the appropriate value of k that simultaneously minimizes the information redundancy and information loss is the maximum value of k for which a plot similar to Figure 5b will not have a local minimum.

In summary, the fuzzy memory system is the set of **T** column nodes with $\overset{*}{\tau}$ values given by Equation 13, with the value of k appropriately matched with c to minimize information redundancy and information loss.

4. Time Series Forecasting

We compare the performance of the self-sufficient fuzzy memory to a shift register in time series forecasting with a few simple illustrations. Our goal here is to illustrate the differences between a simple shift register and the self-sufficient fuzzy memory. Because our interest is in representation of the time series and not in the sophistication of the learning algorithm, we use simple linear regression algorithm to learn and forecast these time series.

We consider three time series with different properties. The first was generated by fractionally integrating white noise (Wagenmakers et al., 2004) in a manner similar to that described in Section 2. The second and third time series were obtained from the online library at http://datamarket.com. The second time series is the mean annual temperature of the Earth from the year 1781 to 1988. The third time series is the monthly average number of sunspots from the year 1749 to 1983 measured from Zurich, Switzerland. These three time series are plotted in the top row of Figure 6. The corresponding two point correlation function of each series is plotted in the middle row of Figure 6. Examination of the two point correlation functions reveal differences between the series. The fractionally-integrated noise series shows long-range correlations falling off like a power law. The temperature series shows correlations near zero (but modestly positive) over short ranges and weak negative correlation over longer times. The sunspots data has both strong positive short-range autocorrelation and a longer range negative correlation, balanced by a periodicity of 130 months corresponding to the 11 year solar cycle.

4.1 Learning and Forecasting Methodology

Let $N_{\max}$ denote the total number of nodes in the memory representation and let N be an index corresponding to each node ranging from 1 to $N_{\max}$. We shall denote the value contained in the nodes at any time step i by $B_i[N]$. The time series was sequentially fed into both the shift register and the self-sufficient fuzzy memory and the representations were evolved appropriately at each time step. The values in the shift register nodes were shifted downstream at each time step as discussed Section 2. At any instant the shift register held information from exactly $N_{\max}$ time steps in the past. The values in the self-sufficient fuzzy memory were evolved as described in Section 3, with $\overset{*}{\tau}$ values taken to be 1, 2, 4, 8, 16, 32,...$2^{(N_{\max}-1)}$, conforming to Equation 13 with $\overset{*}{\tau}_{min} = 1$, $c = 1$ and $k = 8$.

At each time step i, the value from each of the nodes $B_i[N]$ was recorded along with the value of the time series at that time step, denoted by V_i. We used a simple linear regression algorithm to extract the intercept I and the regression coefficients R_N so that the predicted value of the time series

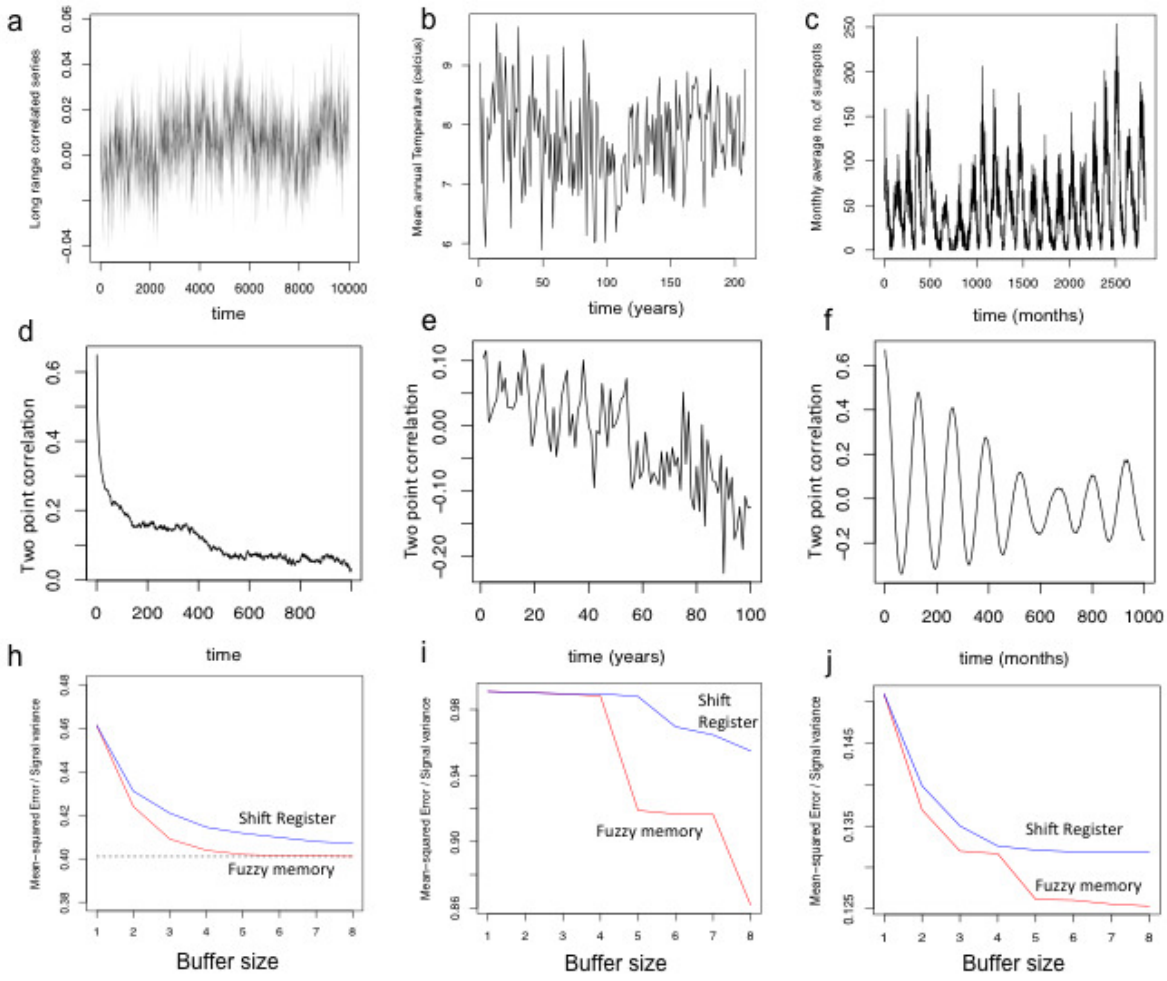


Figure 6: (**a**) Simulated time series with long range correlations based on ARFIMA model with $d = 0.4$, and white noise of standard deviation 0.01. (**b**) Average annual temperature of the Earth from the year 1781 to 1988. (**c**) Monthly average number of sunspots from the year 1749 to 1983. (**d**,**e**,**f**) Two point correlations of the series in **a**, **b** and **c**. (**h**,**i**,**j**) Error in forecasting the series **a**, **b** and **c** using either the fuzzy memory or the the shift register.

at each time step P_i and the squared error in prediction E_i are

$$P_i = I + \sum_{N=1}^{N_{\max}} R_N B_i[N], \qquad E_i = [P_i - V_i]^2.$$

The regression coefficients were extracted by minimizing the total squared error $E = \sum_i E_i$. For this purpose, we used a standard procedure `lm()` in the open source software R.

The accuracy of forecast is inversely related to the total squared error E. To get an absolute measure of accuracy we have to factor out the intrinsic variability of the time series. In the bottom row of Figure 6, we plot the mean of the squared error divided by the intrinsic variance in the time series $\mathrm{Var}[V_i]$. This quantity would range from 0 to 1; the closer it is to zero, the more accurate the prediction.

4.1.1 LONG RANGE CORRELATED SERIES

The long range correlated series (Figure 6a) is by definition constructed to yield a two point correlation that decays as a power law. This is evident from its two point correlation in Figure 6d that is decaying, but always positive. Since the value of the series at any time step is highly correlated with its value at the previous time step, we can expect to generate a reasonable forecast using a single node that holds the value from the previous time step. This can be seen from Figure 6h, where the error in forecast is only 0.46 with a single node. Adding more nodes reduces the error for both the shift register and the self-sufficient fuzzy memory. But for a given number of nodes, the fuzzy memory always has a lower error than the shift register. This can be seen from Figure 6h where the curve corresponding to the fuzzy memory falls below that of the shift register.

Since this series is generated by fractionally integrating white noise, the mean squared error cannot in principle be lower than the variance of the white noise used for construction. That is, there is a lower bound for the error that can be achieved in Figure 6h. The dotted line in Figure 6h indicates this bound. Note that the fuzzy memory approaches this bound with a smaller number of nodes than the shift register.

4.1.2 TEMPERATURE SERIES

The temperature series (Figure 6b) is much more noisy than the long range correlated series, and seems structureless. This can be seen from the small values of its two point correlations in Figure 6e. This is also reflected in the fact that with a small number of nodes, the error is very high. Hence it can be concluded that no reliable short range correlation exist in this series. That is, knowing the average temperature during a given year does not help much in predicting the average temperature of the subsequent year. However, there seems to be a weak negative correlation at longer scales that could be exploited in forecasting. Note from Figure 6i that with additional nodes the fuzzy memory performs better at forecasting and has a lower error in forecasting than a shift register. This is because the fuzzy memory can represent much longer timescales than the shift register of equal size, and thereby exploit the long range correlations that exist.

4.1.3 SUNSPOTS SERIES

The sunspot series (Figure 6c) is less noisy than the other two series considered, and it has an oscillatory structure of about 130 month periodicity. It has high short range correlations, and hence

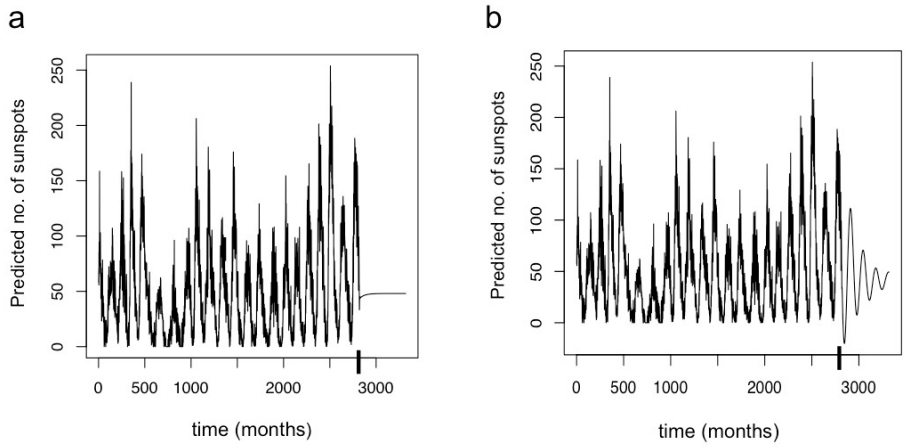


Figure 7: Forecasting the distant future. The sunspots time series of length 2820 is extrapolated for 500 time steps in the future using **(a)** shift register with 8 nodes, and **(b)** fuzzy memory with 8 nodes. The solid tick mark on the x-axis at 2820 corresponds to the point where the original series ends and the predicted future series begins.

even a single node that holds the value from the previous time step is sufficient to forecast with an error of 0.15, as seen in Figure 6j. As before, with more nodes, the fuzzy memory consistently has a lower error in forecasting than the shift register with equal number of nodes. Note that when the number of nodes is increased from 4 to 8, the shift register does not improve in accuracy while the fuzzy memory continues to improve in accuracy.

With a single node, both fuzzy memory and shift register essentially just store the information from the previous time step. Because most of the variance in the series can be captured by the information in the first node, the difference between the fuzzy memory and the shift register with additional nodes is not numerically overwhelming when viewed in Figure 6j. However, there is a qualitative difference in the properties of the signal extracted by the two memory systems. In order to successfully learn the 130 month periodicity, the information about high positive short range correlations is not sufficient, it is essential to also learn the information about the negative correlations at longer time scales. From Figure 6f, note that the negative correlations exist at a timescale of 50 to 100 months. Hence in order to learn this information, these timescales have to be represented. A shift register with 8 nodes cannot represent these timescales but the fuzzy memory with 8 nodes can.

To illustrate that it is possible to learn the periodicity using the fuzzy memory, we forecast the distant future values of the series. In Figure 7, we extend the sunspots series by predicting it for a future of 500 months. The regression coefficients R_N and the intercept I are extracted from the original series of length 2820. For the next 500 time steps, the predictions P_i are treated as actual values V_i, and the memory representations are evolved. Figure 7a shows the series generated using shift register with 8 nodes. The solid tick mark on the x-axis at 2820 represents the point at which

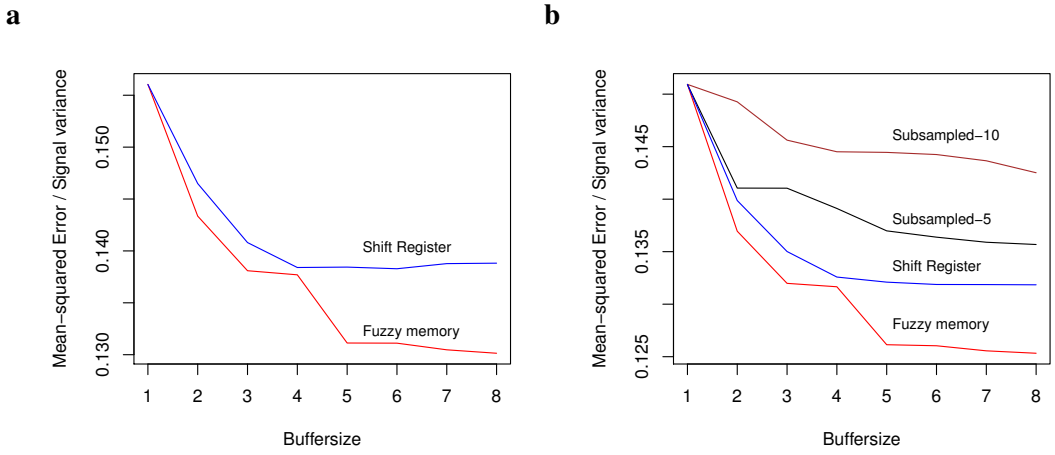


Figure 8: **a.** Testing error in forecasting. The regression coefficients were extracted from the first half of the sunspot series and testing was performed on the second half of the series. **b.** Forecasting error of the fuzzy memory, shift register and subsampled shift register with a node spacing of 5 and 10.

the original series ends and the predicted future series begins. Note that the series forecasted by the shift register immediately settles on the mean value without oscillation. This is because the time scale at which the oscillations are manifest is not represented by the shift register with 8 nodes. Figure 7b shows the series generated by the fuzzy memory with 8 nodes. Note that the series predicted by the fuzzy memory continues in an oscillating fashion with decreasing amplitude for several cycles eventually settling at the mean value. This is possible because the fuzzy memory represents the signal at a sufficiently long time scale to capture the negative correlations in the two-point correlation function.

Of course, a shift register with many more nodes can capture the long-range correlations and predict the periodic oscillations in the signal. However the number of nodes necessary to describe the oscillatory nature of the signal needs to be of the order of the periodicity of the oscillation, about 130 in this case. This would lead to overfitting the data. At least in the case of the simple linear regression algorithm, the number of regression coefficients to be extracted from the data increases with the number of nodes, and extracting a large number of regression coefficients from a finite data set will unquestionably lead to overfitting the data. Hence it would be ideal to use the least number of nodes required to span the relevant time scale.

In order to ensure that the extracted regression coefficients has not overfitted the data, we split the sunspots time series into two halves. We extracted the regression coefficients by using only the first half for training and used the second half for testing the predictions generated by those coefficients. Figure 8a plots this testing error, and should be compared to the training error plotted in Figure 6j. Other than the noticeable fact that the testing error is slightly higher than the training error, the shape of the two plots are very similar for both fuzzy memory and the shift register.

If our goal was to only capture the oscillatory structure of the sunspot series within a small number of regression coefficients, then we could subsample from a lengthy shift register so that

information from both positive and negative correlations can be obtained. Although the subsampled shift register contains relatively few nodes, it cannot self sufficiently evolve in real time; we would need the resources associated with the complete shift register in order to evolve the memory at each moment. By subsampling 8 equidistantly spaced nodes of the shift register 1,11,21,31,...81, and extracting the corresponding regression coefficients, it is possible to extend the series to have an oscillatory structure analogous to Figure 7b. However it turns out that the forecasting error for the subsampled shift register is significantly higher than the forecasting error from the fuzzy memory. Figure 8b shows the forecasting error for subsampled shift register with equidistant node spacing of 5 and 10. Even though the subsampled shift register with a node spacing of 10 extends over a similar temporal range as the fuzzy memory, and captures the oscillatory structure in the data, the fuzzy memory outperforms it with a lower error. The advantage of the fuzzy memory over the subsampled shift register comes from the property of averaging over many previous values at long time scales rather than picking a single noisy value and using that for prediction. This property helps to suppress unreliable fluctuations that could lead to overfitting the data.

5. Discussion

The fuzzy memory holds more predictively relevant information than a shift register with the same number of nodes for long-range correlated signals, and hence performs better in time series forecasting such signals. However, learning the relevant statistics from a lengthy time series is not the same as learning from very few learning trials. To learn from very few learning trials, a learner must necessarily make some generalizations based on some built-in assumptions about the environment. Since the fuzzy memory discards information about the precise time of a stimulus presentation, the temporal inaccuracy in memory can help the learner make such a generalization. Suppose it is useful for a learner to learn the temporal relationship between two events, say A and B. Let the statistics of the world be such that B consistently follows A after a delay period, which on each learning trial is chosen from an unknown distribution. After many learning trials, a learner relying on a shift register memory would be able to sample the entire distribution of delays and learn it precisely. But real world learners may have to learn much faster. Because the fuzzy memory system represents the past information in a smeared fashion, a single training sample from the distribution will naturally let the learner make a scale-free temporal generalization about the distribution of delays between A and B. The temporal profile of this generalization will not in general match the true distribution that could be learned after many learning trials, however the fact that it is available after a single learning trial provides a tremendous advantage for natural learners.

It then seems natural to wonder if human and animal memory resembles the fuzzy memory system. After all, animals have evolved in the natural world where predicting the imminent future is crucial for survival. Numerous behavioral findings on animals and humans are consistent with them having a memory system with scale-free representation of past events (Balsam and Gallistel, 2009; Gallistel and Gibbon, 2000). In human memory studies, the forgetting curve is usually observed to follow a scale invariant power law function (Donkin and Nosofsky, 2012). When humans are asked to reproduce or discriminate time intervals, they exhibit a characteristic scale-invariance in the errors they produce (Rakitin et al., 1998; Wearden and Lejeune, 2008). This is not just a characteristic feature in humans, but in a wide variety of animals like rats, rabbits and pigeons (Roberts, 1981; Smith, 1968). These findings across behavioral tasks and species suggest that a scale-free memory is an adaptive response to a world with structure at many temporal scales.

5.1 Neural Networks with Temporal Memory

Let us now consider the fuzzy memory system in the context of neural networks with temporal memory. It has been realized that neural networks with generic recurrent connectivity can have sufficiently rich dynamics to hold temporal memory of the past. Analogous to how the ripple patterns on a liquid surface contains information about the past perturbations, the instantaneous state of the recurrent network holds the memory of the past which can be simply extracted by training a linear readout layer. Such networks can be implemented either with analog neurons-echo state networks (Jaeger, 2001), or with spiking neurons-liquid state machines (Maass et al., 2002). They are known to be non-chaotic and dynamically stable as long as their spectral radius or the largest eigenvalue of the connectivity matrix has a magnitude less than one. Abstractly, such networks with fixed recurrent connectivity can be viewed as a reservoir of nodes and can be efficiently used for computational tasks involving time varying inputs, including time series prediction (Wyffels and Schrauwen, 2010).

The timescale of these reservoirs can be tuned up by introducing leaky integrator neurons in them (Jaeger et al., 2007). However, a reservoir with finite nodes cannot have memory from infinite past. In fact the criterion for dynamical stability of the reservoir is equivalent to requiring a fading memory (Jaeger, 2002). If we define a memory function of the reservoir to be the precision with which inputs from each past moment can be reconstructed, it turns out that the net memory, or the area under the memory function over all past times, is bounded by the number of nodes in the reservoir $N_{\max}$. The exact shape of the memory function will however depend on the connectivity within reservoir. For a simple shift register connectivity, the memory function is a step function which is 1 up to $N_{\max}$ time steps in the past and zero beyond $N_{\max}$. But for a generic random connectivity the memory function decays smoothly, sometimes exponentially and sometimes as a power law depending on the spectral radius (Ganguli et al., 2008). For linear recurrent networks, it turns out that the memory function is analytically tractable at least in some special cases (White et al., 2004; Hermans and Schrauwen, 2010), while the presence of any nonlinearity seems to reduce the net memory (Ganguli et al., 2008). By increasing the number of nodes in the reservoir, the net memory can be increased. But unless the network is well tailored, as in orthogonal networks (White et al., 2004) or a divergent feed-forward networks (Ganguli et al., 2008), the net memory grows very slowly and sub-linearly with the number of nodes. Moreover, analysis of trajectories in the state-space of a randomly connected network suggests that the net memory will be very low when the connectivity is dense (Wallace et al., 2013).

The self-sufficient fuzzy memory can be viewed as a special case of a linear reservoir with a specific, tailored connectivity. The $\mathbf{t}$ nodes effectively have a diagonal connectivity matrix making them leaky integrators, and the $\mathbf{L}_k^{-1}$ is the linear readout weights that approximately extracts the past inputs. For a white noise input signal, the memory function decays as a power law with exponent -1, and the net memory grows linearly with the number of nodes. However, as described above, the accuracy of reconstruction of the past is not the relevant quantity of interest here, it is the predictive information from the past that is of interest. Scale-free fluctuations in natural world imply that it isn't necessary to be accurate; in fact sacrificing accuracy in a scale-free fashion lets us represent predictive information from exponentially long timescales.

5.2 Unaddressed Issues

Two basic issues essential to real-world machine learning applications have been ignored in this work for the sake of theoretical simplicity. First is that we have simply focused on a scalar time varying signal, while any serious learning problem would involve multidimensional signals. When unlimited storage resources are available, each dimension can be separately represented in a lengthy shift register. To conserve storage resources associated with the time dimension, we could replace the shift register with the self-sufficient fuzzy memory. This work however does not address the issue of conserving storage resources by compressing the intrinsic dimensionality of the signal. In general, most of the information relevant for future prediction is encoded in few combinations of the various dimensions of the signal, called features. Techniques based on information bottleneck method (Creutzig and Sprekeler, 2008; Creutzig et al., 2009) and slow feature analysis (Wiskott and Sejnowski, 2002) can efficiently extract these features. The strategy of representing the time series in a scale invariantly fuzzy fashion could be seen as complementary to these techniques. For instance, slow feature analysis (Wiskott and Sejnowski, 2002) imposes the slowness principle where low-dimensional features that change most slowly are of interest. If the components of a time varying high dimensional signal is represented in a temporally fuzzy fashion rather than in a shift register, then we could potentially extract the slowly varying parts in an online fashion by examining differences in the activities of the largest two $\overset{*}{\tau}$ nodes.

The second issue is that we have ignored the learning and prediction mechanisms while simply focusing on the memory representation. For simplicity we used the linear regression predictor in Section 4. Any serious application should involve the ability to learn nonlinearities. Support vector machines (SVM) adopt an elegant strategy of using nonlinear kernel functions to map the input data to a high dimensional space where linear methods can be used (Vapnik, 1998; Müller et al., 1997). The standard method for training SVMs on time series prediction requires feeding in the data from a sliding time window, in other words providing shift registers as input. It has recently been suggested that rather than using standard SVM kernels on sliding time window, if we used recurrent kernel functions corresponding to infinite recurrent networks, performance can be improved on certain tasks (Hermans and Schrauwen, 2012). This suggests that the gradually fading temporal memory of the recurrent kernel functions is more effective than the step-function memory of shift register used in standard SVMs for time series prediction. Training SVMs with standard kernel functions along with fuzzy memory inputs rather than shift register inputs is an alternative strategy for approaching problems involving signals with long range temporal correlations. Moreover, since $\mathbf{t}$ nodes contain all the temporal information needed to construct the fuzzy memory, directly training the SVMs with inputs from $\mathbf{t}$ nodes could also be very fruitful.

Finally, it should be noted that if our aim is to build an autonomous agent we need both learning and prediction to happen in an online fashion. Many widely-used machine learning algorithms like SVM (Vapnik, 1998) and deep learning networks (Hinton et al., 2006), rely on batch processing which requires the availability of the entire data set prior to learning. Autonomous agents with limited memory resources cannot adopt such learning strategies. The learning mechanism cannot rely on information other than what is instantaneously available in the memory. An online learning algorithm tailored to act on the fuzzy memory representation could potentially be very useful for autonomous agents with finite memory resources.

6. Conclusion

Signals with long-range temporal correlations are found throughout the natural world. Such signals present a distinct challenge to machine learners that rely on a shift-register representation of the time series. Here we have described a method for constructing a self-sufficient scale-free representation of temporal history. The nodes are chosen in a way that minimizes information redundancy and information loss while equally distributing them over all time scales. Although the temporal accuracy of the signal is sacrificed, predictively relevant information from exponentially long timescales is available in the fuzzy memory system when compared to a shift register with the same number of nodes. This could be an extremely useful way to represent time series with long-range correlations for use in machine learning applications.

Acknowledgments

The authors acknowledge support from National Science Foundation grant, NSF BCS-1058937, and Air Force Office of Scientific Research grant AFOSR FA9550-12-1-0369.

Appendix A. Information Redundancy Across Nodes

The information redundancy in the memory representation can be quantified by deriving expressions for mutual information shared between neighboring nodes. When the input signal is uncorrelated or has scale-free long range correlations, it will be shown that the information redundancy is equally spread over all nodes only when the $\overset{*}{\tau}$ values of the nodes are given by Equation 13.

Taking $\mathbf{f}(\tau)$ to be a stochastic signal and the current moment to be $\tau = 0$, the activity of a $\overset{*}{\tau}$ node in the $\mathbf{T}$ column is (see Equation 12)

$$\mathbf{T}(0,\overset{*}{\tau}) = \frac{k^{k+1}}{k!}\int_{-\infty}^{0}\frac{1}{|\overset{*}{\tau}|}\left(\frac{\tau'}{\overset{*}{\tau}}\right)^{k} e^{-k\left(\frac{\tau'}{\overset{*}{\tau}}\right)}\mathbf{f}(\tau')d\tau'.$$

The expectation value of this node can be calculated by simply averaging over $\mathbf{f}(\tau')$ inside the integral, which should be a constant if it is generated by a stationary process. By defining $z = \tau'/\overset{*}{\tau}$, we find that the expectation of $\mathbf{T}$ is proportional to the expectation of $\mathbf{f}$.

$$\left\langle \mathbf{T}(0,\overset{*}{\tau})\right\rangle = \left\langle \mathbf{f}\right\rangle \frac{k^{k+1}}{k!}\int_{0}^{\infty} z^{k}e^{-kz}dz.$$

To understand the information redundancy in terms of correlations among the nodes, we calculate the correlations among the $\mathbf{T}$ nodes when $\mathbf{f}(\tau)$ is either a white noise or a long-range correlated signal.

A.1 White Noise Input

Let $\mathbf{f}(\tau)$ to be white noise, that is $\langle \mathbf{f} \rangle = 0$ and $\langle \mathbf{f}(\tau)\mathbf{f}(\tau') \rangle \sim \delta(\tau - \tau')$. The variance in the activity of each $\overset{*}{\tau}$ node is then given by

$$\begin{aligned}
\langle \mathbf{T}^2(0,\overset{*}{\tau}) \rangle &= \left(\frac{k^{k+1}}{k!}\right)^2 \int_{-\infty}^{0}\int_{-\infty}^{0} \frac{1}{|\overset{*}{\tau}|^2} \left(\frac{\tau}{\overset{*}{\tau}}\right)^k e^{-k\left(\frac{\tau}{\overset{*}{\tau}}\right)} \left(\frac{\tau'}{\overset{*}{\tau}}\right)^k e^{-k\left(\frac{\tau'}{\overset{*}{\tau}}\right)} \langle \mathbf{f}(\tau)\mathbf{f}(\tau') \rangle d\tau d\tau', \\
&= \frac{1}{|\overset{*}{\tau}|} \left(\frac{k^{k+1}}{k!}\right)^2 \int_0^\infty z^{2k} e^{-2kz} dz. \qquad (16)
\end{aligned}$$

As expected, the variance of a large $|\overset{*}{\tau}|$ node is small because the activity in this node is constructed by integrating the input function over a large timescale. This induces an artificial temporal correlation in that node's activity which does not exist in the input function. To see this more clearly, we calculate the correlation across time in the activity of one node, at time τ and τ'. With the definition $\delta = |\tau - \tau'|/|\overset{*}{\tau}|$, it turns out that

$$\langle \mathbf{T}(\tau,\overset{*}{\tau})\ \mathbf{T}(\tau',\overset{*}{\tau}) \rangle = |\overset{*}{\tau}|^{-1} \left(\frac{k^{k+1}}{k!}\right)^2 e^{-k\delta} \sum_{r=0}^{k} \delta^{k-r} \frac{k!}{r!(k-r)!} \int_0^\infty z^{k+r} e^{-2kz} dz. \qquad (17)$$

Note that this correlation is nonzero for any $\delta > 0$, and it decays exponentially for large δ. Hence even a temporally uncorrelated white noise input leads to short range temporal correlations in a $\overset{*}{\tau}$ node. It is important to emphasize here that such temporal correlations will not be introduced in a shift register. This is because, in a shift register the functional value of $\mathbf{f}$ at each moment is just passed on to the downstream nodes without being integrated, and the temporal autocorrelation in the activity of any node will simply reflect the temporal correlation in the input function.

Let us now consider the instantaneous correlation in the activity of two different nodes. At any instant, the activity of two different nodes in a shift register will be uncorrelated in response to a white noise input. The different nodes in a shift register carry completely different information, making their mutual information zero. But in the $\mathbf{T}$ column, since the information is smeared across different $\overset{*}{\tau}$ nodes, the mutual information shared by different nodes is non-zero. The instantaneous correlation between two different nodes $\overset{*}{\tau}_1$ and $\overset{*}{\tau}_2$ can be calculated to be

$$\begin{aligned}
\langle \mathbf{T}(0,\overset{*}{\tau}_1)\ \mathbf{T}(0,\overset{*}{\tau}_2) \rangle &= |\overset{*}{\tau}_2|^{-1} \left(\frac{k^{k+1}}{k!}\right)^2 \int_0^\infty z^{2k} (\overset{*}{\tau}_1/\overset{*}{\tau}_2)^k e^{-kz(1+\overset{*}{\tau}_1/\overset{*}{\tau}_2)} dz, \\
&\propto \frac{(\overset{*}{\tau}_1 \overset{*}{\tau}_2)^k}{(|\overset{*}{\tau}_1| + |\overset{*}{\tau}_2|)^{2k+1}}.
\end{aligned}$$

The instantaneous correlation in the activity of the two nodes $\overset{*}{\tau}_1$ and $\overset{*}{\tau}_2$ is a measure of the mutual information represented by them. Factoring out the individual variances of the two nodes, we have the following measure for the mutual information.

$$I(\overset{*}{\tau}_1,\overset{*}{\tau}_2) = \frac{\langle \mathbf{T}(0,\overset{*}{\tau}_1)\ \mathbf{T}(0,\overset{*}{\tau}_2) \rangle}{\sqrt{\langle \mathbf{T}^2(0,\overset{*}{\tau}_1) \rangle \langle \mathbf{T}^2(0,\overset{*}{\tau}_2) \rangle}} \propto \left[\frac{\sqrt{\overset{*}{\tau}_1/\overset{*}{\tau}_2}}{(1+\overset{*}{\tau}_1/\overset{*}{\tau}_2)}\right]^{2k+1}.$$

This quantity is high when $\overset{*}{\tau}_1/\overset{*}{\tau}_2$ is close to 1. That is, the mutual information shared between neighboring nodes will be high when their $\overset{*}{\tau}$ values are very close.

The fact that the mutual information shared by neighboring nodes is non-vanishing implies that there is redundancy in the representation of the information. If we require the information redundancy to be equally distributed over all the nodes, then we need the mutual information between any two neighboring nodes to be a constant. If $\overset{*}{\tau}_1$ and $\overset{*}{\tau}_2$ are any two neighboring nodes, then in order for $I(\overset{*}{\tau}_1,\overset{*}{\tau}_2)$ to be a constant, $\overset{*}{\tau}_1/\overset{*}{\tau}_2$ should be a constant. This can happen only if the $\overset{*}{\tau}$ values of the nodes are arranged in the form given by Equation 13.

A.2 Long Range Correlated Input

Now consider $\mathbf{f}(\tau)$ such that $\left\langle \mathbf{f}(\tau)\mathbf{f}(\tau')\right\rangle \sim 1/|\tau-\tau'|^{\alpha}$ for large values of $|\tau-\tau'|$. Reworking the calculations analogous to those leading to Equation 17, we find that the temporal correlation is

$$\left\langle \mathbf{T}(\tau,\overset{*}{\tau})\ \mathbf{T}(\tau',\overset{*}{\tau})\right\rangle = \frac{|\overset{*}{\tau}|^{-\alpha}}{2.4^k}\left(\frac{k^{k+1}}{k!}\right)^2 \sum_{r=0}^{k} C_r \int_{-\infty}^{\infty} \frac{|v|^{k-r}}{|v+\delta|^{\alpha}} e^{-k|v|}\, dv.$$

Here $\delta = |\tau-\tau'|/|\overset{*}{\tau}|$ and $C_r = \frac{k!k+r)!}{r!(k-r)!}\frac{2^{k-r}}{(k)^{k+r+1}}$. The exact value of C_r is unimportant and we only need to note that it is a positive number.

For $\alpha > 1$, the above integral diverges at $v = -\delta$, however we are only interested in the case $\alpha < 1$. When δ is very large, the entire contribution to the integral comes from the region $|v| \ll \delta$ and the denominator of the integrand can be approximated as δ^{α}. In effect,

$$\left\langle \mathbf{T}(\tau,\overset{*}{\tau})\ \mathbf{T}(\tau',\overset{*}{\tau})\right\rangle \sim |\overset{*}{\tau}|^{-\alpha}\delta^{-\alpha} = |\tau-\tau'|^{-\alpha}$$

for large $|\tau-\tau'|$. The temporal autocorrelation of the activity of any node should exactly reflect the temporal correlations in the input when $|\tau-\tau'|$ is much larger than the time scale of integration of that node ($\overset{*}{\tau}$). As a point of comparison, it is useful to note that any node in a shift register will also exactly reflect the correlations in the input.

Now consider the instantaneous correlations across different nodes. The instantaneous correlation between two nodes $\overset{*}{\tau}_1$ and $\overset{*}{\tau}_2$ turns out to be

$$\left\langle \mathbf{T}(0,\overset{*}{\tau}_1)\ \mathbf{T}(0,\overset{*}{\tau}_2)\right\rangle = |\overset{*}{\tau}_2|^{-\alpha}\left(\frac{k^{k+1}}{k!}\right)^2 \sum_{r=0}^{k} X_r \frac{\beta^{k-r}(1+\beta^{r-\alpha+1})}{(1+\beta)^{2k-r+1}}. \tag{18}$$

Here $\beta = |\overset{*}{\tau}_1|/|\overset{*}{\tau}_2|$ and each X_r is a positive coefficient. By always choosing $|\overset{*}{\tau}_2| \geq |\overset{*}{\tau}_1|$, we note two limiting cases of interest, $\beta \ll 1$ and $\beta \simeq 1$. When $\beta \ll 1$, the $r = k$ term in the summation of the above equation yields the leading term, and the correlation is simply proportional to $|\overset{*}{\tau}_2|^{-\alpha}$, which is approximately equal to $|\overset{*}{\tau}_2 - \overset{*}{\tau}_1|^{-\alpha}$. In this limit where $|\overset{*}{\tau}_2| \gg |\overset{*}{\tau}_1|$, the correlation between the two nodes behaves like the correlation between two shift register nodes. When $\beta \simeq 1$, note from Equation 18 that the correlation will still be proportional to $|\overset{*}{\tau}_2|^{-\alpha}$. Now if $\overset{*}{\tau}_1$ and $\overset{*}{\tau}_2$ are neighboring nodes with close enough values, we can evaluate the mutual information between them to be

$$I(\overset{*}{\tau}_1,\overset{*}{\tau}_2) = \frac{\left\langle \mathbf{T}(0,\overset{*}{\tau}_1)\ \mathbf{T}(0,\overset{*}{\tau}_2)\right\rangle}{\sqrt{\left\langle \mathbf{T}^2(0,\overset{*}{\tau}_1)\right\rangle\left\langle \mathbf{T}^2(0,\overset{*}{\tau}_2)\right\rangle}} \propto |\overset{*}{\tau}_2/\overset{*}{\tau}_1|^{-\alpha/2}.$$

Reiterating our requirement from before that the mutual information shared by neighboring nodes at all scales should be the same, we are once again led to choose $\overset{*}{\tau}_2/\overset{*}{\tau}_1$ to be a constant which is possible only when the $\overset{*}{\tau}$ values of the nodes are given by Equation 13.

References

R. T. Baillie. Long memory processes and fractional integration in econometrics. *Journal of Econometrics*, 73:5–59, 1996.

P. D. Balsam and C. R. Gallistel. Temporal maps and informativeness in associative learning. *Trends in Neuroscience*, 32(2):73–78, 2009.

J. Beran. *Statistics for Long-Memory Processes*. Chapman & Hall, New York, 1994.

G. Chechik, A. Globerson, N. Tishby, and Y. Weiss. Information bottleneck for gaussian variables. *Journal of Machine Learning Research*, 6:165–188, 2005.

F. Creutzig and H. Sprekeler. Predictive coding and the slowness principle: An information-theoretic approach. *Neural Computation*, 20(4):1026–1041, 2008.

F. Creutzig, A. Globerson, and N. Tishby. Past-future information bottleneck in dynamical systems. *Physical Review E*, 79:041925, 2009.

C. Donkin and R. M. Nosofsky. A power-law model of psychological memory strength in short- and long-term recognition. *Psychological Science*, 23:625–634, 2012.

D. J. Field. Relations between the statistics of natural images and the response properties of cortical cells. *Journal of Optical Society of America A*, 4:2379–2394, 1987.

C. R. Gallistel and J. Gibbon. Time, rate, and conditioning. *Psychological Review*, 107(2):289–344, 2000.

S. Ganguli, D. Huh, and H. Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences of the United States of America*, 105(48):18970–18975, 2008.

D. L. Gilden. Cognitive emissions of $1/f$ noise. *Psychological Review*, 108:33–56, 2001.

C. W. J. Granger and R. Joyeux. An introduction to long-memory time series models and fractional differencing. *Journal of Time Series Analysis*, 1:15–29, 1980.

M. Hermans and B. Schrauwen. Memory in linear recurrent neural networks in continuous time. *Neural Networks*, 23(3):341–355, 2010.

M. Hermans and B. Schrauwen. Recurrent kernel machines: Computing with infinite echo state networks. *Neural Computation*, 24(1):104–133, 2012.

G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

J. R. M. Hosking. Fractional differencing. *Biometrika*, 68(1):165–176, 1981.

H. Jaeger. The echo state approach to analyzing and training recurrent networks. GMD-Report 148, GMD - German National Research Institute for Information Technology, 2001.

H. Jaeger. Short term memory in echo state networks. GMD-Report 152, GMD - German National Research Institute for Information Technology, 2002.

H. Jaeger, M. Lukosevicius, D. Popovici, and U. Siewert. Optimization and applications of echo state networks with leaky integrator neurons. *Neural Networks*, 20:335–352, 2007.

K. Linkenkaer-Hansen, V.V. Nikouline, J. M. Palva, and R. J. Ilmoniemi. Long-range temporal correlations and scaling behavior in human brain oscillations. *Journal of Neuroscience*, 21:1370–1377, 2001.

W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freeman, San Fransisco, CA, 1982.

K. R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In *Proceedings of the International Conference on Analog Neural Networks*, 1997.

E. Post. Generalized differentiation. *Transactions of the American Mathematical Society*, 32:723–781, 1930.

B. C. Rakitin, J. Gibbon, T. B. Penny, C. Malapani, S. C. Hinton, and W. H. Meck. Scalar expectancy theory and peak-interval timing in humans. *Journal of Experimental Psychology: Animal Behavior Processes*, 24:15–33, 1998.

S. Roberts. Isolation of an internal clock. *Journal of Experimental Psychology: Animal Behavior Processes*, 7:242–268, 1981.

K. H. Shankar and M. W. Howard. A scale-invariant internal representation of time. *Neural Computation*, 24:134–193, 2012.

M. C. Smith. CS-US interval and US intensity in classical conditioning of rabbit's nictitating membrane response. *Journal of Comparative and Physiological Psychology*, 66(3):679–687, 1968.

N. Tishby, F. C. Pereira, and W. Bialek. Information bottleneck method. In *Proceedings of 37th Allerton Conference on Communication and Computation*, Monticello, IL, 1999.

A. Torralba and A. Oliva. Statistics of natural image categories. *Network: Computation in Neural Systems*, 14(3):391–412, 2003.

G. C. Van Orden, J. G. Holden, and M. T. Turvey. Self organization of cognitive performance. *Journal of Experimental Psychology: General*, 132:331–350, 2003.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

R. F. Voss and J. Clarke. $1/f$ noise in music and speech. *Nature*, 258:317–318, 1975.

E. J. Wagenmakers, S. Farrell, and R. Ratcliff. Estimation and interpretation of $1/f^{\alpha}$ noise in human cognition. *Psychonomic Bulletin & Review*, 11(4):579–615, 2004.

E. Wallace, H. R. Maei, and P. E. Latham. Randomly connected networks have short temporal memory. *Neural Computation*, 25:1408–1439, 2013.

J. H. Wearden and H. Lejeune. Scalar properties in human timing: conformity and violations. *Quarterly Journal of Experimental Psychology*, 61:569–587, 2008.

O. L. White, D. D. Lee, and H. Sompolinsky. Short-term memory in orthogonal neural networks. *Physical Review Letters*, 92(14):148102, 2004.

L. Wiskott and T. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.

F. Wyffels and B. Schrauwen. A comparative study of reservoir computing strategies for monthly time series prediction. *Neurocomputing*, 73:1958–1964, 2010.

Journal of Machine Learning Research 14 (2013) 3813-3817 Submitted 4/13; Revised 9/13; Published 12/13

BudgetedSVM: A Toolbox for Scalable SVM Approximations

Nemanja Djuric NEMANJA@TEMPLE.EDU
Liang Lan LANLIANG@TEMPLE.EDU
Slobodan Vucetic VUCETIC@TEMPLE.EDU
304 Wachman Hall, Temple University
1805 North Broad Street
Philadelphia, PA 19122, USA

Zhuang Wang ZJWANG@US.IBM.COM
Global Business Services, IBM
1475 Phoenixville Pike
West Chester, PA 19380, USA

Editor: Antti Honkela

Abstract

We present BudgetedSVM, an open-source C++ toolbox comprising highly-optimized implementations of recently proposed algorithms for scalable training of Support Vector Machine (SVM) approximators: Adaptive Multi-hyperplane Machines, Low-rank Linearization SVM, and Budgeted Stochastic Gradient Descent. BudgetedSVM trains models with accuracy comparable to LibSVM in time comparable to LibLinear, solving non-linear problems with millions of high-dimensional examples within minutes on a regular computer. We provide command-line and Matlab interfaces to BudgetedSVM, an efficient API for handling large-scale, high-dimensional data sets, as well as detailed documentation to help developers use and further extend the toolbox.

Keywords: non-linear classification, large-scale learning, SVM, machine learning toolbox

1. Introduction

Support Vector Machines (SVMs) are among the most popular and best performing classification algorithms. Kernel SVMs deliver state-of-the-art accuracies on non-linear problems, but are characterized by linear growth in the number of support vectors with data size, which may prevent learning from truly large data. In contrast, linear SVMs cannot capture non-linear concepts, but are very scalable and allow learning from large data with limited resources. Aimed at bridging the representability and scalability gap between linear and non-linear SVMs, recent advances in large-scale learning resulted in powerful algorithms that enable scalable training of non-linear SVMs, such as Adaptive Multi-hyperplane Machines (AMM) (Wang et al., 2011), Low-rank Linearization SVM (Zhang et al., 2012), and Budgeted Stochastic Gradient Descent (BSGD) (Wang et al., 2012). With accuracies comparable to kernel SVM, the algorithms are scalable to millions of examples, having training and inference times comparable to linear and orders of magnitude shorter than kernel SVM.

We present BudgetedSVM, an open-source C++ toolbox for scalable non-linear classification. The toolbox provides an Application Programming Interface (API) for efficient training and testing of non-linear classifiers, supported by data structures designed for handling data which cannot fit in memory. BudgetedSVM can be seen as a missing link between LibLinear and LibSVM (Hsieh et al., 2008; Chang and Lin, 2011), combining the efficiency of linear with the accuracy of kernel SVM.

We also provide command-line and Matlab interfaces, providing users with an efficient, easy-to-use tool for large-scale non-linear classification.

2. Non-linear Classifiers for Large-scale Data

Before taking a closer look at the implementation and usage details of the BudgetedSVM toolbox, in this section we give a brief description of the implemented algorithms.

2.1 Adaptive Multi-hyperplane Machines (AMM)

Wang et al. (2011) proposed a classifier that captures non-linearity by assigning a number of linear hyperplanes to each of C classes from a set $\mathcal{Y}$. Given a D-dimensional example $\mathbf{x}$, the AMM multi-class classifier has the following form,

$$f(\mathbf{x}) = \arg\max_{i \in \mathcal{Y}} g(i,\mathbf{x}), \text{ where } g(i,\mathbf{x}) = \max_{j=1,\ldots,b_i} \mathbf{w}_{ij}^{\mathrm{T}}\mathbf{x}, \tag{1}$$

where the i^{th} class is assigned b_i weight vectors with the total budget $B = \sum_i b_i$. AMM is learned via Stochastic Gradient Descent (SGD). The hyper-parameters include a regularization parameter λ, the number of training epochs e, the maximum number of non-zero weights per class B_{lim}, $b_i \leq B_{lim}$, and weight pruning parameters k (pruning frequency) and c (pruning aggressiveness). As an initial guideline to the users, we experimentally found that for most data sets the values $e = 5$ (or $e = 1$ for very large data), $B_{lim} = 50$, $k = 10{,}000$, and $c = 10$ are appropriate choices, leaving only λ to be determined by cross-validation.

When $b_1, \ldots, b_C$ are fixed to 1, the AMM model reduces to linear multi-class SVM (Crammer and Singer, 2002), and the learning algorithm is equivalent to Pegasos, a popular linear SVM solver (Shalev-Shwartz et al., 2007). As it is a widely-used linear SVM solver, we also provide the Pegasos algorithm directly as a shortcut in the BudgetedSVM toolbox.

2.2 Low-rank Linearization SVM (LLSVM)

Zhang et al. (2012) proposed to approximate kernel SVM optimization by a linear SVM using low-rank decomposition of the kernel matrix. The approximated optimization is solved via Dual Coordinate-Descent (DCD) (Hsieh et al., 2008). The binary classifier has the form

$$f(\mathbf{x}) = \operatorname{sign}\big(\mathbf{w}^{\mathrm{T}}(\mathbf{M} \cdot g(\mathbf{x}))\big),$$

where $g(\mathbf{x}) = [k(\mathbf{x},\mathbf{z}_1), \ldots, k(\mathbf{x},\mathbf{z}_B)]^{\mathrm{T}}$, $\{\mathbf{z}_i\}_{i=1,\ldots,B}$ is a set of landmark points of size B, $k(\mathbf{x},\mathbf{z}_i)$ is a kernel function, $\mathbf{w}$ defines a separating hyperplane in the linearized kernel space (found using the DCD method), and $\mathbf{M}$ is a $B \times B$ mapping matrix. The hyper-parameters include kernel parameters, regularization parameter λ, and the number of landmark points B. Parameter B controls a trade-off between speed and accuracy, while kernel parameters and λ are best determined by cross-validation.

2.3 Budgeted Stochastic Gradient Descent (BSGD)

Wang et al. (2012) proposed a budgeted algorithm which maintains a fixed number of support vectors in the model, and incrementally updates them during the SGD training. The multi-class BSGD

	Pegasos	AMM	LLSVM	BSGD	RBF-SVM
Training time	$O(NCS)$	$O(NSB)$	$O(NSB^2+NSB)$	$O\big(N(C+S)B\big)$	$O(INCS)$
Prediction time	$O(CS)$	$O(SB)$	$O(SB^2+SB)$	$O\big((C+S)B\big)$	$O(NCS)$
Model size	$O(CD)$	$O(DB)$	$O(DB+B^2)$	$O\big((C+D)B\big)$	$O(NCS)$

Table 1: Time and space complexities of the classification algorithms

classifier has the same form as (1), but with $g(i,\mathbf{x})$ defined as

$$g(i,\mathbf{x}) = \sum_{j=1}^{B} \alpha_{ij} k(\mathbf{x},\mathbf{z}_j),$$

where $\{\mathbf{z}_j\}_{j=1,\ldots,B}$ is the support vector set, and α_{ij} is a class-specific parameter associated with the j^{th} support vector. We implemented Pegasos-style training, where the budget is maintained through either merging (where RBF kernel is used) or random removal of support vectors. The hyper-parameters include the number of epochs e, kernel parameters, regularization parameter λ, and budget size B. Parameters B and e control a speed-accuracy trade-off, while kernel parameters and λ are best determined by cross-validation.

2.4 Time and Space Complexity

Time and space complexities of the algorithms are summarized in Table 1, where N is the number of training examples, C is the number of classes, D is the data dimensionality, data sparsity S is the average number of non-zero features, and B is the model size for AMM, BSGD, and LLSVM. Parameter I for SVM with RBF kernel (RBF-SVM) denotes a number of training iterations, empirically shown to be super-linear in N (Chang and Lin, 2011).

3. The Software Package

BudgetedSVM can be found at `http://www.dabi.temple.edu/budgetedsvm/`. The software package provides a C++ API, comprising functions for training and testing of non-linear models described in Section 2. Each model can be easily trained and tested by calling the corresponding *train/predict* function, defined in `mm_algs.h`, `bsgd.h`, and `llsvm.h` header files. The API also provides functions for handling large-scale, high-dimensional data, defined in `budgetedSVM.h` file.

BudgetedSVM sequentially loads data chunks into memory to allow large-scale training, storing to memory only indices and values of non-zero features as a linked list. Furthermore, implementation of sparse vectors is optimized for high-dimensional data, allowing faster kernel computations and faster updates of hyperplanes and support vectors than linked list (e.g., as in LibSVM) or array implementation of vectors (e.g., as in MSVMpack by Lauer and Guermeur, 2011) used for regular-scale problems, where either time or memory costs can become prohibitively large during training in a large-scale setting. In particular, vectors are split into disjoint chunks where pointers to each chunk are stored in an array, and memory for a chunk is allocated only if one of its elements is non-zero. While significantly reducing time costs, we empirically found that this approach incurs very limited memory overhead even for data with millions of features. Consequently, BudgetedSVM vector reads and writes are performed memory-efficiently in constant time. Moreover, by storing and incrementally updating support vector ℓ_2-norms after each training step, time to compute popular kernels (e.g., linear, Gaussian, polynomial) scales only linearly with sparsity S. Further implementation details can be found in a comprehensive developer's guide.

	Pegasos		AMM			LLSVM			BSGD			RBF-SVM	
Data set	e.r.	t.t.	e.r.	*B*	t.t.	e.r.	*B*	t.t.	e.r.	*B*	t.t.	e.r.	t.t.
webspam						3.46	500	2.5m	2.04	500	2.0m		
$N = 280{,}000$	7.94	0.5s	4.74	9	3s	2.60	1,000	6.1m	1.72	1,000	3.9m	0.77	4.0h
$D = 254$						1.99	3,000	0.5h	1.49	3,000	0.2h	(#SV: 26,447)	
rcv1						4.97	500	0.2h	3.33	500	0.8h		
$N = 677{,}399$	2.73	1.5s	2.39	19	9s	4.23	1,000	0.5h	2.92	1,000	1.5h	2.17	20.2h
$D = 47{,}236$						3.05	3,000	2.2h	2.53	3,000	4.4h	(#SV: 50,641)	
mnist8m-bin						6.84	500	1.6h	2.23	500	2.3h		
$N = 8{,}000{,}000$	22.71	1.1m	3.16	18	4m	4.59	1,000	3.8h	1.92	1,000	4.9h	0.43	N/A[1]
$D = 784$						2.59	3,000	15h	1.62	3,000	16.1h		

Table 2: Error rates (e.r.; in %) and training times[2](t.t.) on benchmark data sets

We also provide command-line and Matlab interfaces for easier use of the toolbox, which follow the user-friendly format of LibSVM and LibLinear. For example, we can type `budgetedsvm-train -A 1 a9a_train.txt a9a_model.txt` in the command prompt to train a classifier on the *adult9a* data set. The `-A 1` option specifies that we use the AMM algorithm, while the data is loaded from `a9a_train.txt` file and the trained model is stored to the `a9a_model.txt` file. Similarly, we type `budgetedsvm-predict a9a_test.txt a9a_model.txt a9a_output.txt` to evaluate the trained model, which loads the testing data from `a9a_test.txt` file, the model from `a9a_model.txt` file, and stores the predictions to `a9a_output.txt` file. We also provide a short tutorial which outlines the basic steps for using the BudgetedSVM interfaces.

3.1 Performance Comparison

The BudgetedSVM toolbox can learn an accurate model even for data with millions of examples and features, with training times orders of magnitude faster than RBF-SVM trained using LibSVM. For illustration, in Table 2 we give comparison of error rates and training times on binary classification tasks using several large-scale data sets (Wang et al., 2011). On *webspam* and *rcv1* it took LibSVM hours to train RBF-SVM, while BudgetedSVM algorithms with much smaller budgets achieved high accuracy within minutes, and even seconds in the case of AMM. Similarly, RBF-SVM training on large-scale *mnist8m-bin* could not be completed in a reasonable time on our test machine, while the implemented algorithms were trained within a few hours on extremely limited budgets to achieve low error rates. More detailed analysis of the BudgetedSVM algorithms can be found in their respective papers.

4. Conclusion and Future Work

BudgetedSVM implements four large-scale learners. Using optimized functions and data structures as a basis, through our and community efforts we plan to add more classifiers, such as Tighter Perceptron (Wang and Vucetic, 2009) and BPA (Wang and Vucetic, 2010), to make BudgetedSVM a more inclusive toolbox of budgeted SVM approximations.

Acknowledgments

This work was supported by NSF grants IIS-0546155 and IIS-1117433.

1. Listed accuracy was obtained after 2 days of P-packSVM training on 512 processors (Zhu et al., 2009).
2. We excluded data loading time (evaluated on Intel® E7400 with 2.80GHz processor, 4GB RAM).

References

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2002.

C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *International Conference on Machine Learning*, pages 408–415, 2008.

F. Lauer and Y. Guermeur. MSVMpack: A multi-class support vector machine package. *Journal of Machine Learning Research*, 12:2293–2296, 2011.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *International Conference on Machine Learning*, pages 807–814, 2007.

Z. Wang and S. Vucetic. Tighter perceptron with improved dual use of cached data for model representation and validation. In *International Joint Conference on Neural Networks*, pages 3297–3302, 2009.

Z. Wang and S. Vucetic. Online passive-aggressive algorithms on a budget. In *International Conference on Artificial Intelligence and Statistics*, pages 908–915, 2010.

Z. Wang, N. Djuric, K. Crammer, and S. Vucetic. Trading representability for scalability: Adaptive multi-hyperplane machine for nonlinear classification. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2011.

Z. Wang, K. Crammer, and S. Vucetic. Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale SVM training. *Journal of Machine Learning Research*, 13:3103–3131, 2012.

K. Zhang, L. Lan, Z. Wang, and F. Moerchen. Scaling up kernel SVM on limited resources: A low-rank linearization approach. In *International Conference on Artificial Intelligence and Statistics*, 2012.

Z. A. Zhu, W. Chen, G. Wang, C. Zhu, and Z. Chen. P-packSVM: Parallel primal gradient descent kernel SVM. In *IEEE International Conference on Data Mining*, 2009.