Regularized factor models

by

Martha White

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

University of Alberta

Abstract

This dissertation explores **regularized factor models** as a simple unification of machine learning problems, with a focus on algorithmic development within this known formalism. The main contributions are (1) the development of generic, efficient algorithms for a subclass of regularized factorizations and (2) new unifications that facilitate application of these algorithms to problems previously without known tractable algorithms. Concurrently, the generality of the formalism is further demonstrated with a thorough summary of known, but often scattered, connections between supervised and unsupervised learning problems and algorithms.

The dissertation first presents the main algorithmic advances: convex reformulations of non-convex regularized factorization objectives. A convex reformulation is developed for a general subset of regularized factor models, with an efficiently computable optimization for five different regularization choices. The thesis then describes advances using these generic convex reformulation techniques in three important problems: multi-view subspace learning, semi-supervised learning and estimating autoregressive moving average models. These novel settings are unified under regularized factor models by incorporating problem properties in terms of regularization. Once expressed as regularized factor models, we can take advantage of the convex reformulation techniques to obtain novel algorithms that produce global solutions. These advances include the first global estimation procedure for two-view subspace learning and for autoregressive moving average models. The simple algorithms obtained from these general convex reformulation techniques are empirically shown to be effective across these three problems on a variety of datasets.

This dissertation illustrates that many problems can be specified as a simple regularized factorization, that this class is amenable to global optimization and that it is advantageous to represent machine learning problems as regularized factor models.

Acknowledgements

I cannot give enough thanks to my two amazing advisors, Michael Bowling and Dale Schuurmans. They have spent countless hours discussing research and life with me, and it has been immeasurably important to me.

I would like to thank the Department of Computing Science and everyone in it. Many people, still at the department or now scattered across the world, have made my PhD more full and created an amazing environment for research.

I would like to thank NSERC, Alberta Innovates, the Killam Program and the University of Alberta for funding throughout my PhD. Without their generous support of this research, it could not have occurred.

Finally, I would like to thank my husband, Adam White, my partner in research and life.

Table of Contents

List of Tables

List of Figures

List of Notations

$\mathbb{R}^+$ The set of all non-negative real numbers, including infinity: $[0, \infty]$ 7

$\equiv$ Indicates that two optimizations have equivalent solution sets: if $\operatorname{argmin}_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \operatorname{argmin}_{\boldsymbol{\theta}} g(\boldsymbol{\theta})$, then write $\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \equiv \min_{\boldsymbol{\theta}} g(\boldsymbol{\theta})$ 7

∇ The gradient operator. 7

∂ The subgradient operator. 7

$\dagger$ The pseudo-inverse of a matrix. 7

F A strictly convex potential function that defines a natural exponential family distribution, $p_F(\mathbf{z}|\boldsymbol{\theta}) = \exp(\mathbf{z}'\boldsymbol{\theta} - F(\boldsymbol{\theta}))p_0(\mathbf{z})$. 12

$\mathbf{f}$ A transfer function, $\mathbf{f} = \nabla F$, for a given potential function F, that is used to define the corresponding Bregman divergence $D_F(\hat{\mathbf{z}}||\mathbf{z}) = F(\hat{\mathbf{z}}) - F(\mathbf{z}) - \mathbf{f}(\mathbf{z})'(\hat{\mathbf{z}} - \mathbf{z})$. 12

$L(\cdot, X)$ A generic loss function for a given data matrix, X. 38

$\|\mathbf{x}\|$ A generic norm on a vector $\mathbf{x}$ (norms are always convex). 38

$\|\mathbf{y}\|^*$ The conjugate norm: $\|\mathbf{y}\|^* = \max_{\|\mathbf{x}\| \leq 1} \mathbf{x}'\mathbf{y}$, where $\|\mathbf{x}\|^{**} = \|\mathbf{x}\|$. 38

$\|\mathbf{x}\|_p$ A p-norm, $1 \leq p \leq \infty$, $\|\mathbf{x}\|_p = (\sum_i |\mathbf{x}_i|^p)^{1/p}$ with $\|\mathbf{x}\|_p^* = \|\cdot\|_q$ for $\frac{1}{p} + \frac{1}{q} = 1$. 38

$\|X\|$ A generic matrix norm on X (norms are always convex). 38

$\|Y\|^*$ Conjugate norm of $\|X\|$, where $\|Y\|^* = \max_{\|X\| \leq 1} \operatorname{tr}(X'Y)$ and $\|X\|^{**} = \|X\|$. 38

$\|X\|_{(p,q)}$ The *induced* (p, q) *norm* on X: $\|X\|_{(p,q)} = \max_{\|\mathbf{z}\|_p \leq 1} \|X\mathbf{z}\|_q$, defined in (Horn and Johnson, 1990, §5.6). 38

$\|X\|_{(\mathcal{C},\diamond)}$ The *generalized induced norm* on X: $\|X\|_{(\mathcal{C},\diamond)} = \max_{\mathbf{z} \in \mathcal{C}} \|X\mathbf{z}\|_\diamond$ for any bounded closed set $\mathcal{C} \subset \mathbb{R}^n$ such that $\operatorname{span}(\mathcal{C}) = \mathbb{R}^n$ (see Lemma 3) 38

$\|X\|_{sp}$ The *spectral norm*: $\|X\|_{sp} = \|X\|_{(2,2)} = \sigma_{max}(X)$. 38

$\|X\|_{tr}$ The *trace norm* or *nuclear norm*, conjugate of the spectral norm: $\|X\|_{tr} = \sum_i \sigma_i(X)$. . 38

$\|X\|_F$ The *Frobenius norm*: $\|X\|_F = \sqrt{\operatorname{tr}(X'X)} = \sqrt{\sum_i \sigma_i^2(X)}$. 38

$\|X\|_{r,s}$ The *block norm*, $\|X\|_{r,s} = (\sum_i (\sum_t |X_{ij}|^r)^{\frac{s}{r}})^{\frac{1}{s}}$. 38

$\|X\|_{r,s}^*$ Conjugate of block norm: $\|X\|_{r,s}^* = \|X\|_{r^*,s^*}$ such that $\frac{1}{r} + \frac{1}{r^*} = \frac{1}{s} + \frac{1}{s^*} = 1$. . . 38

Chapter 1

Introduction

Statistical machine learning involves learning functional relationships using sampled data from a population. For example, one might be interested in predicting energy output from a solar panel based on current weather information, i.e., finding a function f from input variables to target variables. Typically, we cannot iterate the entire space of possible input and output variables; rather we obtain *samples* from that space to learn an approximate functional relationship that generalizes to unseen instances. In addition to predictive functions, another goal may be to extract structure, such as mapping instances to groups or finding novel representations, such as lower-dimensional manifolds.

Due to the ubiquitous need for analyzing gathered data in a wide variety of fields, statistical machine learning algorithms have been repeatedly reinvented, often with slightly different perspectives or procedural approaches. For example, several multiple linear regression techniques were introduced separately but are all equivalent, including simultaneous linear predictions and redundancy analysis in psychology and reduced rank regression in statistics. As another example, several graph-based dimensionality-reduction techniques were introduced separately, including Isomap, Laplacian eigenmaps and locally linear embeddings; these techniques, however, are actually variants of kernel principal components analysis. This proliferation of techniques has led to difficulties in understanding, particularly in terms of algorithm selection and interpreting algorithm properties. Moreover, algorithmic advances are slowed by repeated development, and generalizations to novel settings can be unclear when the algorithm is coupled with the specific problem setting.

In recent years, there has been a focus on understanding connections between these algorithms in a unified way. Many pairwise connections have been recognized, such as the connection between principal components analysis and k-means clustering (Jong and Kotz, 1999) or the connection between canonical correlation analysis and linear discriminant analysis (Bach and Jordan, 2006). In addition, there have been several unification formalisms proposed, namely as generalized eigenvalue

problems (Borga et al., 1997; Kokiopoulou et al., 2011), least-squares optimizations (De la Torre, 2012) and generalized linear model optimizations (Gordon, 2003; Banerjee et al., 2005; Singh and Gordon, 2008). A common theme among many of the unifications is obtaining a *factorization*: a basis C and representation Φ that best explains the data X, typically in the relationship $X = \mathbf{f}(C\Phi)$ for some transfer function $\mathbf{f}$.

A unified view of matrix factorization was recently introduced by Singh and Gordon (2008) that encompasses many of these previous formalisms. This view encompasses many algorithms, including non-negative matrix factorization, principal components analysis, probabilistic latent semantic indexing and Bregman co-clustering to name a few and facilitated the development of a general Newton projection algorithm. In this thesis, I propose to explore a modest generalization to this formalism, which I call **regularized factor models**. Though there are more general unifications, described in Section 3.3, I propose to explore this unifying formalism because it

1. reduces the number of modeling choices to a small set of choices
2. appears to unify a large section of machine learning algorithms, despite its simplicity
3. is amenable to efficient optimization for several important cases
4. has an intuitive maximum likelihood interpretation; and
5. provides transparent assumptions on the data and problem properties.

1.1 Objective

This thesis aims to further understanding on the following question:

> What *unifications* and *algorithmic development* for statistical machine learning problems can be achieved under regularized factor models?

Unification under regularized factor models should have the focus of separating the problem from the algorithm. Often, the problem and algorithmic solution can be conflated, and a focus on unifying under a formalism with precise modeling choices promotes a clear separation. This separation enables clear extensions on the problem, as well as clarifying assumptions behind previous algorithms and when they transfer to new settings. The objective is not necessarily to champion regularized factor models as a "theory of everything", but rather to promote useful insights into how to set modeling choices for new problems and exploit clear connections to extend or reuse algorithms to solve those problems.

My focus in terms of **algorithmic development** is finding principled techniques that are guaranteed to find global solutions to the specified problem. Without guarantees, tuning parameters and interpreting results can be difficult. For example, one could run an algorithm for a range of meta-parameters (such as regularization parameters); if the algorithm is not guaranteed to return global solutions, returning instead local minima or ad hoc solutions, it is not possible to distinguish if the differing solutions are due to the change in meta-parameter setting or from different local minima. To make precise problem specifications practically useful, guarantees that the problem can be solved are essential. Though precise problem specifications often make their own approximations to the true problem, one can more clearly identify the modeling choices that are inaccurate or are relaxations, whereas it is often difficult to interpret the consequence of local solutions.

1.2 Approach

An important factor in our approach for algorithmic advances under regularized factorization is a focus on *convex reformulations*. Convex optimizations guarantee that all local minima are actually global minima: if an efficient descent approach is available, then a global solution to the specified objective can be obtained. The approach centers around identifying non-convex objectives that can be transformed in to convex objectives. This approach benefits from the fact that with proper choices in problem specification we need not lose any modeling power but gain in terms of optimization. Though convex reformulations are not the only approach for obtaining global solutions, (1) they enable generalization, in that one reformulation insight extends to many settings, due to properties of convex functions and (2) there are many algorithms for solving convex optimization problems, adding another level of versatility.

To enable further unification, I shift focus to incorporating more problem properties in terms of regularization. Many previous algorithms did not use regularization, but rather focus on data re-weighting and kernelization to obtain useful properties. A modern approach is to incorporate structure using regularizers, such as the ℓ_1 regularizer for sparsity. In addition to generalizing the encodable structure, there are also useful distributional properties encoded by the regularizers. Finally, this shift is not only key for incorporating novel problem settings, including multi-view subspace learning, semi-supervised learning and autoregressive moving average models, but also for enabling algorithmic advances. In particular, norm-based regularizers are convex, making them amenable to convex reformulation techniques.

1.3 Contributions

The key contributions of this dissertation are the following.

Convex formulations for regularized factor models. I have co-developed a convex reformulation for a general subset of regularized factor models with an efficiently computable optimization for five cases. The five cases are for different regularization and constraint set choices; the remaining modeling choices remain general, where any convex loss function can be used. The approach involves deriving an induced norm on a joint variable, given the regularizers chosen for the regularized factor model. In certain cases, a closed form recovery procedure of the factorized variables from the joint variable is provided. For the other cases, a generic boosting strategy is given. This work was published in a refereed conference proceeding (Zhang et al., 2011).

A tractable formulation of multi-view subspace learning. This thesis illustrates that multi-view subspace learning, including canonical correlation analysis, can be cast as a regularized factor model, and shows that the problem can be convexly reformulated. This contribution improves upon the above work for the partitioned $(2, 1)$-block norm, and provides an improved boosting recovery algorithm. In addition to a careful analysis in terms of runtimes, and objective values on synthetic data, improved performance is achieved over the state-of-the-art in image denoising. This work was published in a refereed conference proceeding (White et al., 2012).

A formulation of semi-supervised learning with a theoretical justification for the addition of unlabeled data. This thesis establishes a connection between supervised and unsupervised learning through the use of regularized factor models. This connection is exploited to develop a principled semi-supervised learning algorithm for which we can make a variance reduction argument. An experimental analysis reveals the usefulness of selecting modeling choices under a unified regularized factor model formalism, which benefits from obvious extensions and enables a surprising level of generality in problem properties using only choices of kernels, instance weights and non-linear transfers. This work was published first for least-squares losses (Xu et al., 2009) and then generalized to Bregman divergences (White and Schuurmans, 2012).

A tractable formulation of semi-supervised learning. This thesis contributes a convex reformulation for a different form of semi-supervised learning. This form consists of simultaneously learning a new representation for the input data and using that new representation to predict the labeled data. This reformulation is similar to the convex reformulation for multi-view learning, but

with a different form for the two views. The experimental results indicate that this convex semi-supervised learning algorithm outperforms four other semi-supervised learning algorithms on six real datasets. This work was published in a refereed conference proceeding (Zhang et al., 2011).

A tractable maximum likelihood estimation algorithm for autoregressive moving average models. Finally, this thesis illustrates that with a small relaxation, we could exploit the distributional assumptions behind certain regularizers to derive a convex reformulation of autoregressive moving average models. In particular, this work tackles high-dimensional vector time series using autoregressive moving average models for multi-step forecasting, which has been under-explored due to difficulties in algorithm development and iterated prediction. Empirical results on a broad range of algorithms from time series and subspace identification indicate that the proposed approach produces more stable forecasts and is significantly faster than many previous approaches. This work will appear in a refereed conference proceeding (White et al., 2015).

1.4 Overview

This thesis is structured to first describe regularized factor models then useful background concepts for supervised and unsupervised learning, while illustrating how they are regularized factor models. Then, convex reformulation approaches are described followed by three chapters using the reformulations to obtain important algorithmic advances in subspace learning, semi-supervised learning and autoregressive moving average models, respectively.

Chapter 2 - Background: regularized factor models

This chapter presents background on regularized factor models. In particular, it first explains the maximum likelihood approach underlying regularized factor models and justifies the choice of factorizing for finding a unifying theory. Then exponential family distributions and Bregman divergences are introduced, and the connection between them explained.

Chapter 3 - Preliminary insights: supervised and unsupervised learning using factor models

This chapter describes how both supervised learning and unsupervised learning can be represented as a regularized factorization. This chapter contains some small contributions to unifying supervised and unsupervised learning. The primary goal, however, is to introduce standard constructs in machine learning, such as kernels and instance weighting, as well as to illustrate that many classical problems and algorithms in supervised and unsupervised learning are unified under regularized

factor models. The in-depth analysis on several supervised and unsupervised learning algorithms gives more intuition on how to represent machine learning problems as regularized factorizations. The chapter concludes with a discussion of previous unification frameworks.

Chapter 4 - Convex formulations for regularized factor models

This chapter describes how to tackle the regularized factorization optimization by using *convex reformulations*. The general reformulation approach is described, where the product $Z = C\Phi$ is directly learned by characterizing the induced norm on Z given constraints and regularizers on C and Φ. Then, a convex reformulation is given for a general class of regularized factor model problems. Finally, though the induced norm may exist and be convex for the general class, it is not always practical to compute. Therefore, we also provide five cases where the induced norm is practically computable, which are useful for finding tractable formulations for the problems in the following three chapters.

Chapter 5 - Subspace learning and sparse coding using regularized factor models

This chapter presents three important unsupervised learning problems that can be convexly reformulated: sparse coding, single-view subspace learning and multi-view subspace learning. The regularized factor models objective defined for each of the problems is first motivated for each problem; then the solution approaches described in Chapter 4 are applied to solve each of these objectives. The chapter concludes with an empirical study showing the advantages of the convex multi-view algorithm over the (suboptimal) alternating solver and over formulating the problem as a single-view subspace learning problem.

Chapter 6 - Semi-supervised learning using regularized factor models

This chapter describes my algorithmic contributions to semi-supervised learning for the two typical approaches to semi-supervised learning: using unlabeled data to improve weights learned directly on the input data and using unlabeled data to learn a new representation and learn weights on that "improved" representation using only the labeled data. I first illustrate that the forward-reverse prediction equivalence that enables supervised learning to be formalized as a regularized factor model that facilitates the development of a principled semi-supervised learning algorithm. In particular, it is principled because the unlabeled data reduces the variance of the error estimate, indicating provable benefits of the addition of unlabeled data. Then, I provide a convex semi-supervised learning algorithm for the second type of semi-supervised learning, using the advances in Chapter 4. Experimental analysis is provided for both semi-supervised learning approaches.

Chapter 7 - Autoregressive moving average models using regularized factor models

This chapter describes a novel global maximum likelihood algorithm for autoregressive moving average models. A similar approach to previous convex reformulations is given; however, the approach is re-derived for non-i.i.d. data and for the case that the desired distribution is only parametrized by the factor weights, rather than by both weights and factors. Finally, an experimental analysis is given versus non-convex autoregressive moving average solutions and recent method-of-moments algorithms for state-space models.

Chapter 8 - Perspectives and future work

This work concludes with a broader perspective on the impacts of the research and a discussion of important research directions.

Chapter 2

Background: regularized factor models

In this chapter, we discuss how the objective of maximum likelihood and maximum a posterior (MAP) estimation leads to the regularized factor models objective:

$$\min_{C \in \mathcal{C} \subset \mathbb{R}^{n \times k}} \min_{\Phi \in \mathcal{F} \subset \mathbb{R}^{k \times T}} L\left(C\Phi\ ;\ X\right) + R_k(C, \Phi) \tag{2.1}$$

where

1. $X \in \mathbb{R}^{n \times T}$ is the data matrix with T samples of an n-dimensional vector, $X = [X_{:1}, \ldots, X_{:T}]$.
2. $\mathcal{C} \subset \mathbb{R}^{n \times k}$ is the set from which the basis C is chosen.
3. $\mathcal{F} \subset \mathbb{R}^{k \times T}$ is the set from which the representation Φ is chosen.
4. $L(\cdot, X_{:t})$ is any convex loss function in the first argument, where for convenience, the loss is overloaded for all the data: $L(\cdot, X) : \mathbb{R}^{n \times T} \to \mathbb{R}$.
5. $k \in \{1, 2, \ldots\}$ is the dimension of the representation, and can be infinite; typically, this value is fixed *a priori*, but in general, need not be and can be controlled by the regularizer.
6. $R_k : \mathcal{C} \times \mathcal{F} \to \mathbb{R}^+$ is the regularizer on the factors. This regularizer might be additively decomposable into two separate regularizers, $R_{C,k} : \mathbb{R}^{n \times k} \to \mathbb{R}^+$ on the basis and $R_{\Phi,k} : \mathbb{R}^{k \times T} \to \mathbb{R}^+$ on the representation.

The above optimization assumes the regularizer weights are part of the regularizer function. Certain regularizers can be cast as constraints and vice versa; this equivalence is discussed in Appendix A.2. Both regularizers and constraint sets are included in the loss since for most situations this relationship is ambiguous and there is not a clear preference for one or the other.

The specification of this optimization is simple, with only two choices that can be modified to produce different problems: the choice of convex loss on the multiplication of the factors, $C\Phi$,

and the choice of constraints and regularizers on C and Φ that imposes certain properties on these factors. Interestingly, however, as I will show in this thesis, the variability in these two choices encompasses a large number of problems in machine learning and provides a simple, unified way to examine both the problem and potential algorithms. Before getting into specific problem instances in later chapters, in this chapter I will give some intuition for why this simple objective has such expressive power. Simultaneously, mathematical notation and distributional assumptions required for the remainder of the thesis will be introduced and explained.

2.1 Why regularized factor models?

In this section, we see how the regularized factor models optimization is obtained from the general problem of maximizing the likelihood of parameters, given data, and the ubiquitous goal in many fields of finding "explanatory" factors. Assume that you have a data matrix, $X \in \mathbb{R}^{n \times T}$, with T samples of n-dimensional vectors $X_{:t}$ with density $p(X_{:t}|\boldsymbol{\theta})$ for some unknown parameters $\boldsymbol{\theta}$. For example, if the data contains the heights and weights of T patients in a hospital, then we might believe that each patient's height and weight $X_{:t} \in \mathbb{R}^2$ is drawn from a Gaussian distribution parameterized by unknown mean $\boldsymbol{\mu}$ and unknown covariance Σ in height and weight.

A general approach to finding these parameters is to select the parameters that maximize the likelihood of the data:

$$\max_{\boldsymbol{\theta}} p(X_{:1}, \ldots, X_{:T}|\boldsymbol{\theta}).$$

If the data is i.i.d.[1] as is typically assumed, then we can rewrite this optimization as:

$$\max_{\boldsymbol{\theta}} \log p(X_{:1}, \ldots, X_{:T}|\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \log \left(p(X_{:1}|\boldsymbol{\theta}) \cdots p(X_{:T}|\boldsymbol{\theta})\right) \equiv -\min_{\boldsymbol{\theta}} \sum_{t=1}^{T} \log p(X_{:t}|\boldsymbol{\theta})$$

where $\equiv$ means that the $\operatorname{argmax} = \operatorname{argmin}$, though the objective values themselves may not equal.[2] A more general goal than maximizing the likelihood of the data is to maximize the posterior probability of the parameters given the data, i.e. the maximum a posterior (MAP) estimate:

$$\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|X_{:1}, \ldots, X_{:T}) = \max_{\boldsymbol{\theta}} p(X_{:1}, \ldots, X_{:T}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \equiv \min_{\boldsymbol{\theta}} -\sum_{t=1}^{T} \log p(X_{:t}|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta})$$

using Bayes' rules for the first equality. The density $p(\boldsymbol{\theta})$ specifies the prior on $\boldsymbol{\theta}$. The MAP estimate is equivalent to maximizing likelihood if a non-informative (i.e., uniform) prior is assumed on $\boldsymbol{\theta}$; otherwise, the prior enables known or desired properties to be specified on the unknown parameters.

[1] In Chapter 7, we discuss the change in this maximization if the data is temporally related; the remaining analysis carries through simply, since we still obtain densities over each sample, $X_{:t}$.

[2] Note also that $\max_{\boldsymbol{\theta}} p(X_{:1}, \ldots, X_{:T}|\boldsymbol{\theta}) \equiv \max_{\boldsymbol{\theta}} \log p(X_{:1}, \ldots, X_{:T}|\boldsymbol{\theta})$ because $\log$ is a monotonically increasing function on $[0, 1]$, giving equivalent argmax solutions.

For certain distributions and priors, the negative log of the density has a nice form in this optimization. Arguably the most common distribution assumed on the data is the Gaussian distribution, which results in the simple Euclidean, least-squares loss:

$$\min_{\boldsymbol{\theta}} -\sum_{t=1}^{T} \log p(X_{:t}|\boldsymbol{\mu}, \Sigma = I) \equiv \min_{\boldsymbol{\theta}} \sum_{t=1}^{T} \|X_{:t} - \boldsymbol{\theta}\|_2^2.$$

Though this prototypical loss is often chosen to facilitate optimization, many other distributions similarly result in *convex losses*. Convex losses are preferable as they guarantee that all local minima are actually *global* minima; non-convex losses, on the other hand, are prone to local minima and there is no guarantee on how close any such local minimum is to the true solution of the objective. Importantly, for the general class of exponential family distributions, the negative log of the density has a convex form, called a *Bregman divergence*. The unit-variance Gaussian distribution is an example of an exponential family, with a corresponding Bregman divergence equal to the Euclidean loss. Other notable exponential family, Bregman divergence pairs include the Poisson distribution and unnormalized KL-divergence; the Bernoulli distribution and log-loss; and the multinomial distribution, generalized to continuous values between $[0, 1]$ and the relative entropy. See Section 2.2 for how to obtain this equivalence between maximizing likelihood of exponential families and minimizing Bregman divergences, as well as a more complete table of exponential families and their Bregman divergences in Table A.1.

The second component of the MAP optimization is the prior; to maintain a convex optimization, the prior density on $\boldsymbol{\theta}$ should be *log-concave*. A function is log-concave if the logarithm of the function is concave; therefore, the negative log is convex. The set of log-concave distributions is large, including any exponential family distribution and convolutions of log-concave distributions. See Section A.2 for the connection between regularizers and constraints.

Putting these components together, for exponential family density on $X_{:t}$, corresponding Bregman divergence L, and convex regularizer $R(\boldsymbol{\theta}) = -\log p(\boldsymbol{\theta})$, we obtain the following MAP optimization:

$$\min_{\boldsymbol{\theta}} -\sum_{t=1}^{T} \log p(X_{:t}|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) \equiv \min_{\boldsymbol{\theta}} \sum_{t=1}^{T} L\left(\boldsymbol{\theta}\ ;\ X_{:t}\right) + R(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} L\left(\boldsymbol{\theta}\ ;\ X\right) + R(\boldsymbol{\theta})$$

where for simplicity of notation, we overload L to also mean the sum over all samples. If $\boldsymbol{\theta} = C$ only, then we get traditional, fixed size parameters; if $\boldsymbol{\theta} = C\Phi$, then our parameters change with the number of samples.

The remaining question is the form of the distribution parameters. Theoretically, this form is not limited to any class; in practice, however, we consider simpler forms that are amenable to optimization and, in some cases, amenable to interpretation. Interestingly, across fields as different as

psychometrics, product management and natural language processing with varying terms such as factor analysis, representation learning and latent variable models, the parameter for the distribution is often considered to consist of a weighting, C, of k factors, $\boldsymbol{\phi}_t \in \mathbb{R}^k$ for each data point $X_{:t}$. Assuming this form, the distribution can be parameterized in two ways: by both the weighting and the factors $\boldsymbol{\theta} = (C, \Phi)$ or solely by the weighting $\boldsymbol{\theta} = C$. For the majority of the thesis, I assume that $\boldsymbol{\theta} = (C, \Phi)$; Section 2.1.1 and Chapter 7 includes a discussion on choosing a parametrization that consists only of the weightings, $\boldsymbol{\theta} = C$, and shows that we can arrive at the same parametrization $\boldsymbol{\theta} = (C, \Phi)$ by *maximizing* over the "nuisance" parameter Φ rather than *marginalizing*.

For the joint prior $p(\boldsymbol{\theta} = (C, \Phi))$, a natural goal is to specify separate properties on the basis and representation. In general, for regularized factor models, we simply assume that $R_k(C, \Phi) = -\log p(C, \Phi)$. If we assume, however, that C and Φ are independent *a priori*, then $p(C, \Phi) = p(C)p(\Phi)$. The log-likelihood then separates into two functions, since $\log p(\boldsymbol{\theta}) = \log(p(C)p(\Phi)) = \log p(C) + \log p(\Phi)$ where now $R_{C,k}(C) = -\log p(C)$ and $R_{\Phi,k}(\Phi) = -\log p(\Phi)$. It may also be possible that $\log p(C, \Phi)$ is separable, even if C and Φ are not independent, again allowing separate properties to be placed on the two free variables. If joint properties are desired, typically the properties would be on the product, $p(C\Phi)$.

The resulting MAP optimization now becomes

$$\min_{\boldsymbol{\theta}} -\sum_{t=1}^{T} \log p(X_{:t}|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) \equiv \min_{C \in \mathbb{R}^{n \times k}} \min_{\Phi \in \mathbb{R}^{k \times T}} L\left(C\Phi\ ;\ X\right) + R_{C,k}(C) + R_{\Phi,k}(\Phi). \quad (2.2)$$

We can see, therefore, that if the problem requires explanatory factors and a maximum likelihood approach, the regularized matrix factorization loss aptly defines the objective. In Section 3.2, we see that many unsupervised learning problems can be cast as a regularized factor models. Moreover, in the following Chapters 5, 6 and 7, I present algorithmic advances for several machine learning problems that are encompassed by this objective, namely, multi-view subspace learning, semi-supervised learning and autoregressive moving average models.

2.1.1 Different choices for the distribution parameters

So far, we have assumed that $\boldsymbol{\theta} = (C, \Phi)$; it is also valid, however, to assume that the distribution is parameterized only by the weights C. As we will see is the case for autoregressive moving average models in Chapter 7, the goal is to parametrize by only C, but conditioning on the factors Φ makes the distribution simpler to optimize. Since the distribution $p(X|C)$ can be difficult to handle in the optimization (see Section 7.1 for an example), we want to write the optimization still in terms of the simpler $p(X|C, \Phi)$. This conditioning enables the application of the expectation-maximization algorithm, which introduces the variable Φ by marginalizing over it, as in the next lemma.

Lemma 1. *For an auxiliary density $q(\cdot)$ over Φ, and entropy $H(q(\cdot))$, it follows that*

$$\log p(X|C) = \log \int p(X, \Phi|C)\, d\Phi = \max_{q(\cdot)} \int q(\Phi) \log p(X, \Phi|C)\, d\Phi + H(q(\cdot)).$$

The proof is given in Appendix G.1.

The maximum likelihood problem can now be re-expressed as

$$\min_{C} \min_{\{q(\cdot)\}} - \int q(\Phi) \log p(X, \Phi|C)\, d(\Phi) - H(q(\cdot)) - \log p(C), \tag{2.3}$$

where in a standard EM algorithm, the M step would consist of optimizing C given $\{q(\cdot)\}$, and the E step would consist of (implicitly) optimizing $\{q(\cdot)\}$ given C (Neal and Hinton, 1998).

A standard variant of the log likelihood in (2.3) can then be obtained simply by dropping the entropy regularizer $H(q(\cdot))$. This modification leads to the minimization selecting a Dirac delta distribution on Φ that selects the minimum Φ and a far simpler formulation, sometimes known as "hard EM" or "Viterbi EM" (Brown et al., 1993):

$$\begin{aligned}\min_{C} \min_{\Phi} - \log p(X, \Phi|C) - \log p(C) &= \min_{C} \min_{\Phi} - \log p(X|\Phi, C) - \log p(\Phi|C) - \log p(C) \\ &= \min_{C} \min_{\Phi} - \log p(X|\Phi, C) - \log p(C, \Phi).\end{aligned}$$

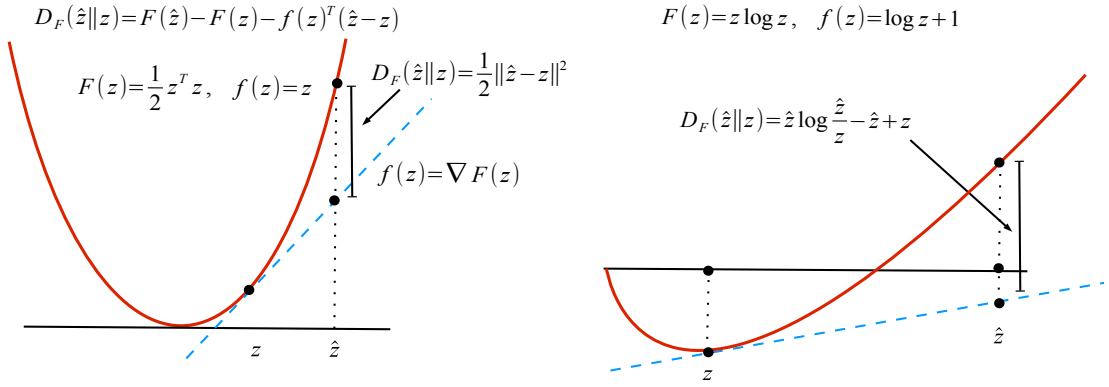
The above follows from the chain rule which states that $P(X, Y) = P(X|Y)P(Y)$, giving the decomposition $p(X, \Phi|C) = p(X|\Phi, C)p(\Phi|C)$ and $p(\Phi|C)p(C) = p(C, \Phi)$.

Even for this different parametrization, therefore, using this entropy simplification, we arrive at the same optimization as when we assumed that the parameters contained both weightings and factors, $\boldsymbol{\theta} = (C, \Phi)$. Throughout the text, therefore, we focus on this setting; in Chapter 7, this choice will be discussed again when the data is not i.i.d.

2.2 Bregman divergences and exponential family distributions

Another important modeling choice is the loss function between the factors $C\Phi$ and X. In this section, I describe the relationship between Bregman divergences and exponential family distributions that enables intuitive choices of loss functions. In addition to the choice of transfer for the Bregman divergence/exponential family, in Section 3.2, I describe some additions to these loss functions, including kernels and instance weighting, that further clarify possible modeling choices for the loss function in regularized factor models. The reader who is already familiar with Bregman divergences can safely skip this section.

Under maximum likelihood, a distribution over the data is chosen with unknown parameters, where the goal is to learn those parameters. Despite the fact that often Gaussian distribution are



(a) Euclidean loss

(b) Relative entropy loss

Figure 2.1: Two Bregman divergences with different transfer functions. The function pictured is the potential function, F. The Bregman divergence corresponds to the difference between the value at the point, $\hat{\mathbf{z}}$, and the first order Taylor expansion around $\mathbf{z}$ **(a)** For the identity transfer, $\mathbf{f}(\mathbf{z}) = \mathbf{z}$, the Bregman divergence corresponds to the Euclidean loss. **(b)** For the logarithmic transfer shifted by 1, $\mathbf{f}(\mathbf{z}) = \mathbf{z}\log(\mathbf{z})$, the Bregman divergence corresponds to the relative entropy or normalized KL-divergence.

chosen for $p(X|\boldsymbol{\theta})$, the set of distributions that result in convex optimizations is actually much more general, i.e., **exponential family distributions**. This generality allows both different data properties to be expressed, such as skew or heavy-tailedness, as well as different non-linear transfers to be applied to the factorization explaining the data. Below, I describe how exponential family distributions have a corresponding convex loss, called a **Bregman divergence**, and explain how this connection also enables non-linear transfers. Table A.1 describes properties of different exponential family distributions and lists their corresponding Bregman divergence.

For any strictly convex differentiable **potential function** $F : \mathbb{R}^k \to \mathbb{R}$, where $\mathbf{f} = \nabla F$ is the associated **transfer function**, the Bregman divergence between two vectors, $\hat{\mathbf{z}}$ and $\mathbf{z}$ is given by

$$D_F(\hat{\mathbf{z}}||\mathbf{z}) = F(\hat{\mathbf{z}}) - F(\mathbf{z}) - f(\mathbf{z})'(\hat{\mathbf{z}} - \mathbf{z}).$$

Intuitively, the Bregman divergence is the difference between F at $\hat{\mathbf{z}}$ and the first order Taylor expansion around $\mathbf{z}$ evaluated at $\hat{\mathbf{z}}$. This divergence, therefore, measures the remaining difference of Taylor order two and higher. For example, for $F(\mathbf{z}) = \frac{1}{2}\mathbf{z}'\mathbf{z}$, the remainder after differencing $F(\hat{\mathbf{z}})$ and its first order Taylor expansion is the second order $\frac{1}{2}(\hat{\mathbf{z}}-\mathbf{z})'\mathbf{g}(F)(\mathbf{z})(\hat{\mathbf{z}}-\mathbf{z}) = \frac{1}{2}\|\hat{\mathbf{z}}-\mathbf{z}\|_2^2$ since the Hessian $\mathbf{g}(F)(\mathbf{z}) = I$ and any higher order derivatives are zero. Figure 2.1 illustrates this intuition for two common Bregman divergences.

Bregman divergences have many useful properties as a loss, including

i) **Coincidence axiom:** $D_F(\hat{\mathbf{z}}||\mathbf{z}) = 0 \Leftrightarrow \hat{\mathbf{z}} = \mathbf{z}$

ii) **Non-negativity:** $D_F(\hat{\mathbf{z}}||\mathbf{z}) \geq 0$

iii) **Convexity:** D_F is convex in the first argument

iv) **Linearity:** For strictly convex, differentiable potential functions F_1 and F_2, then $D_{F_1+\alpha F_2} = D_{F_1} + \alpha D_{F_2}$

Bregman divergences encompass a wide range of losses, including least squares (identity transfer), cross entropy or unnormalized KL-divergence (sigmoidal transfer) and relative entropy or KL-divergence (softmax transfer) (Kivinen and Warmuth, 2001).

I now show the connection to natural exponential family distributions, arising from the fact that exponential families are similarly defined using a potential function F.

Definition 1. *An **exponential family distribution** has probability density function (or probability mass function for discrete distributions) defined as*

$$p(\mathbf{z}|\boldsymbol{\theta}) = \exp(T(\mathbf{z})'\eta(\boldsymbol{\theta}) - F(\boldsymbol{\theta}))p_0(\mathbf{z})$$

*for given functions η, T, F and p_0. A **natural exponential family distribution** or **regular exponential family distribution** has η and T as identity functions,*

$$p_F(\mathbf{z}|\boldsymbol{\theta}) = \exp(\mathbf{z}'\boldsymbol{\theta} - F(\boldsymbol{\theta}))p_0(\mathbf{z})$$

Regular exponential family distributions are a general class of distributions with useful properties. A few distributions, such as the Beta distribution and the lognormal distribution, are exponential family but not regular exponential family distributions. Many distributions, however, are regular exponential family distributions, including the Gaussian, gamma, chi-square, beta, Weibull, Bernoulli and Poisson distributions; see Table A.1 for a more complete list of regular exponential family distributions and their properties.

The connection between Bregman divergences and regular exponential family distributions becomes clear when maximizing the likelihood of data. If the function F defining the regular exponential family is a strictly convex, differentiable function, then

$$\begin{aligned}
\min_{\boldsymbol{\theta}} -\log p_F(\mathbf{z}|\boldsymbol{\theta}) &= \min_{\boldsymbol{\theta}} -\log p_0(\mathbf{z}) - \mathbf{z}'\boldsymbol{\theta} + F(\boldsymbol{\theta}) && \triangleright \text{ by definition} \\
&\equiv \min_{\boldsymbol{\theta}} -\mathbf{z}'\boldsymbol{\theta} + F(\boldsymbol{\theta}) && \triangleright -\log p_0(\mathbf{z}) \text{ does not affect the min} \\
&\equiv \min_{\boldsymbol{\theta}} F(\boldsymbol{\theta}) - \mathbf{z}'\boldsymbol{\theta} - F(\mathbf{z}) + \mathbf{z}\mathbf{f}^{-1}(\mathbf{z}) && \triangleright -F(\mathbf{z}) + \mathbf{z}\mathbf{f}^{-1}(\mathbf{z}) \\
&= \min_{\boldsymbol{\theta}} F(\boldsymbol{\theta}) - F(\mathbf{z}) - \mathbf{z}'(\boldsymbol{\theta} - \mathbf{f}^{-1}(\mathbf{z})) && \text{does not affect the min} \\
&= \min_{\boldsymbol{\theta}} D_F(\boldsymbol{\theta} \,||\, \mathbf{f}^{-1}(\mathbf{z})) && \triangleright \text{ by definition}
\end{aligned} \tag{2.4}$$

Similarly, the KL-divergence between two natural exponential family distributions with the same potential but different parameters, $p_F(\cdot|\boldsymbol{\theta}_1)$ and $p_F(\cdot|\boldsymbol{\theta}_2)$, can be represented as a Bregman divergence between their parameters: $KL(p_F(\cdot|\boldsymbol{\theta}_1)||p_F(\cdot|\boldsymbol{\theta}_2)) = D_F(\boldsymbol{\theta}_2 \;||\; \boldsymbol{\theta}_1)$. See Banerjee et al. (2005, pg. 1741) for more details.

Surprisingly, every regular exponential family distribution has a strictly convex, differentiable F function, giving the relationship in (2.4). In fact, a bijection has been established between regular exponential families and regular Bregman divergences (Banerjee et al., 2005), defined below. This link corresponds to the fact that F is often called the *log-normalizer* or *log-partition function*.

Definition 2. *A continuous function $g : \Theta \to \mathbb{R}^+$ is an **exponentially convex function** iff it is a Laplace transform of a nonnegative finite measure. Examples include $\exp(ax + bx^2)$, $(\exp(x) - 1)/x$, x^{-n} and $\sinh(x)/x$ (Ehm et al., 2003). For g an exponentially convex function on an open set Θ such that $F(\boldsymbol{\theta}) = \ln(g(\boldsymbol{\theta}))$ is strictly convex, then D_F is called a **regular Bregman divergence**.*[3]

(Banerjee et al., 2005, Theorem 6) *There is a bijection between regular exponential families and regular Bregman divergences.*

Therefore, for any regular exponential family, there is a corresponding unique regular Bregman divergence that is guaranteed to be convex in its first argument. This property of regular exponential family distributions will be crucial for developing tractable algorithms to learn the parameters $\boldsymbol{\theta}$.

[3] Originally in (Banerjee et al., 2005), they define the regular Bregman divergence to be the conjugate Bregman divergence, D_{F^*}, which is described in Section 3.1. From Lemma 2, these two are interchangeable, and so both can be defined as regular Bregman divergences.

Chapter 3

Preliminary insights: supervised and unsupervised learning using factor models

This chapter provides required background on supervised and unsupervised learning used in the remainder of the thesis, as well as describing several small contributions I have made to these fields. The goal of this chapter is to illustrate that supervised learning and unsupervised learning can both be represented as regularized factor models, introducing also standard constructs in machine learning, such as kernels and instance weighting. The developments in this chapter elucidate that many classical problems and algorithms in supervised and unsupervised learning are unified under regularized factor models; this unification is mostly achieved through different choices of losses, instance weights and kernels. The following chapters expand this set, by focusing on different regularization choices. The preliminary insights developed in this chapter both clarify how later problems can be represented as regularized factor models, and introduces the reverse-prediction framework that enables the development of a principled semi-supervised learning algorithm in Chapter 6. The chapter concludes with a discussion on previous unification frameworks, which is more clearly presented after understanding how problems and algorithms are unified under regularized factor models.

3.1 Supervised learning using factor models

Supervised learning is a fundamental machine learning problem: given pairwise data, infer a function from an input vector to an output or target vector. Learning is executed on labeled data: pairwise examples of input vectors and the desired output value (i.e., supervisory signal) are used to interpolate a function. This section illustrates that supervised learning can actually be formulated as a regularized factor model, where the representation Φ is set to the target labels and only the basis C is optimized for the factorization $C\Phi$.

The key behind establishing this formulation is through the connection between *forward pre-*

diction and *reverse prediction*. In linear forward prediction, given input data X and target labels Y, the goal is to learn weights W such that $WX \approx Y$. In linear reverse prediction, the (somewhat counterintuitive) goal is to learn weights C such that $CY \approx X$. Interestingly, a one-to-one mapping between W and C can be established, meaning that one can equivalently choose between the typical forward prediction optimization or the reverse prediction (regularized factorization) optimization. In addition to novel insights, this connection also enables the development of a novel principled semi-supervised learning algorithm, described in Chapter 6.

The connection between forward and reverse prediction is first described for the simpler least-squares (Gaussian) setting, and then extended to general exponential families. The goal of this section is not only to describe supervised learning as a regularized factor model, but also to introduce the reader to many standard techniques in supervised and unsupervised learning, such as kernels and instance weighting, that will be useful for later developments.

3.1.1 Linear forward prediction

Assume we are given input data as a matrix $X \in \mathbb{R}^{n \times T}$, where each column is an n-dimensional instance.[1] Assume we are given a $k \times T$ matrix of prediction targets Y. For example, the data matrix X may have n attributes for T patients, where there might be one prediction target $k = 1$, such as patient weight (regression problem), a yes or no if the patient will survive cancer treatment (hard classification), or the probability of developing diabetes (soft classification). Regression and classification problems can be represented similarly, with additional constraints for classification. For hard classification, $Y \in \{0, 1\}^{k \times T}$ such that $\mathbf{1}Y = \mathbf{1}$ (a single 1 in each column). For soft classification, $Y \in [0, 1]^{k \times T}$ such that $\mathbf{1}Y = \mathbf{1}$ (each column sums to 1). Throughout, we assume that the data, X, has been centered, i.e. $X = \tilde{X} - \frac{1}{T}(\sum_{t=1}^{T} \tilde{X}_{:t})\mathbf{1}'$ for the original data $\tilde{X}$.

A common form of supervised learning is to find a $k \times n$ matrix W such that $WX \approx Y$, obtained by minimizing the least squares loss, which is the squared Frobenius norm $\|\cdot\|_F$ (i.e., maximizing likelihood of X assuming each sample $X_{:t}$ is Gaussian distributed):

$$\min_W \sum_{t=1}^{T} \|WX_{:t} - Y_{:,t}\|_2^2 = \min_W \|WX - Y\|_F^2 = \min_W \operatorname{tr}\left((WX - Y)'(WX - Y)\right). \quad (3.1)$$

This convex minimization is easily solved to obtain the global minimizer $W = YX^\dagger$, for $X^\dagger = X'(XX')^{-1}$ the pseudo-inverse of X, assuming that X is full rank and $n \leq T$ to ensure the inverse

[1] Though it is typical in supervised learning to have instances as rows rather than columns, the analysis is equivalent and clarifies the connection to unsupervised learning, where instances are often viewed as columns.

of XX' exists. To see why, first expand

$$\begin{aligned}\operatorname{tr}\left((WX-Y)'(WX-Y)\right) &= \operatorname{tr}\left(X'W'WX\right) - \operatorname{tr}\left(X'W'Y\right) - \operatorname{tr}\left(Y'WX\right) + \operatorname{tr}\left(Y'Y\right) \\ &= \operatorname{tr}\left(X'W'WX\right) - 2\operatorname{tr}\left(X'W'Y\right) + \operatorname{tr}\left(Y'Y\right).\end{aligned}$$

Using the circular property of trace, $\operatorname{tr}(X'W'Y) = \operatorname{tr}(Y'WX)$, and gradients of the trace norm[2],

$$\begin{aligned}&\nabla \operatorname{tr}\left((WX-Y)'(WX-Y)\right) = 2WXX' - 2YX' = 0 \\ &\implies W = YX'(XX')^{-1} = YX^{\dagger}\end{aligned}$$

where the gradient is with respect to W and XX' is invertible because $T \geq n$ and $\operatorname{rank}(X) = n$. The result can then be used to make predictions on test data via $\hat{\mathbf{y}} = W\mathbf{x}$, where for classification, $\hat{\mathbf{y}}$ can be thresholded or learning can be done with a sigmoidal transfer discussed in Section 2.2.

Linear least squares can be extended to incorporate regularization, kernels and instance weighting. **Regularization** is a general tool in supervised learning to reduce overfitting to the training data and *a priori* enforce properties on the learned models, such as sparsity (see Sections 2.1 and 5.2). Some regularizers permit a closed form solution, such as ridge regularization

$$\min_{W} \operatorname{tr}\left((WX-Y)'(WX-Y)\right) + \alpha \operatorname{tr}\left(WW'\right) \tag{3.2}$$

yielding the altered solution $W = YX'(XX' + \alpha I)^{-1}$ since $\nabla \operatorname{tr}(WW') = 2W$. The regularizer weight, $\alpha > 0$, controls the influence of the regularizer. General regularization functions R do not permit a closed form solution; however, if the regularizer has a subgradient, $\partial R(W)$, then the optimization can be solved with a (non smooth) optimization method, using the subgradient $\nabla \operatorname{tr}\left((WX-Y)'(WX-Y)\right) + \alpha \partial R(W) = 2WXX' - 2YX' + \alpha \partial R(W)$.

Kernels map the data into a new space using similarities between nonlinearly transformed data points; learning in the kernel space is a linear learning problem, but corresponds to non-linear learning in the original space. More formally, a *kernel function* $K_{\mathbf{x}} : \mathbb{R}^n \to \mathbb{R}$, from a Reproducing Kernel Hilbert Space, returns a similarity measure from a given point, $\mathbf{x}$. Common examples include the linear kernel, $K_{\mathbf{x}}(\mathbf{y}) = \mathbf{x}^T\mathbf{y}$, the Gaussian kernel, $K_{\mathbf{x}}(\mathbf{y}) = \exp(-||\mathbf{x}-\mathbf{y}||_2^2/2\sigma^2)$, and a polynomial kernel, $K_{\mathbf{x}}(\mathbf{y}) = (\mathbf{x}^T\mathbf{y} + c)^d$. The *kernel matrix*, which summarizes the similarities between the data points $K_{ij} = K(\mathbf{x}_t, \mathbf{x}_j)$, $K \succeq 0$, replaces the data matrix, where the new features are distances (see Figure 3.1(a)), given certain conditions are met under the *representer theorem* (Kimeldorf and Wahba, 1971).

For a least-squares loss, a kernelized version of (3.2) can be easily derived from the identity $X'(XX' + \alpha I)^{-1} = (X'X + \alpha I)^{-1}X'$, since this implies the solution of (3.2) can be expressed

[2] See (Petersen, 2004) for useful matrix inequalities and operations.

as $W = AX'$ for $A = Y(X'X + \alpha I)^{-1}$. By learning A instead of W, the input data X need only appear in the problem through the inner product matrix $X'X \in \mathbb{R}^{T\times T}$. Once the input data appears only as inner products, positive definite kernels can be used to obtain non-linear prediction models (Schölkopf and Smola, 2002). For least squares, the kernelized training problem can then be expressed as a loss on A with $\text{tr}\left((AX'X - Y)'(AX'X - Y)\right) + \alpha \text{ tr}\left(AX'XA'\right)$; in general, for K some implicit feature representation of $X'X$, such as a Gaussian kernel, we get

$$\min_A \text{ tr}\left((AK - Y)'(AK - Y)\right) + \alpha \text{ tr}\left(A'AK\right). \tag{3.3}$$

It is easy to verify that $A = Y(K + \alpha I)^{-1}$ is the global minimizer. Given this solution, test predictions can be made via $\hat{\mathbf{y}} = A\mathbf{k}$ where $\mathbf{k}$ corresponds to the implicit inner products $X'\mathbf{x}$.

Finally, we can consider adding **instance weighting** to the loss. To express weighting, let $\Lambda \in \mathbb{R}^{T\times T}$ be a diagonal matrix of strictly positive instance weights. Then the previous training problem can be expressed as

$$\min_A \|(AK - Y)\Lambda\|_F^2 + \alpha \text{ tr}\left(A'AK\right) = \min_A \text{ tr}((AK - Y)'(AK - Y)\Lambda^2) + \alpha \text{ tr}\left(A'AK\right)$$

with the optimal solution $A = Y\Lambda(K\Lambda + \alpha I)^{-1}$.

3.1.2 Linear forward-reverse prediction connection

The previous results can all be replicated in the reverse direction, where one attempts to predict inputs X from targets Y. In particular, given optimal solutions to reverse prediction problems, the corresponding forward solutions can be recovered exactly. Although reverse prediction might seem counterintuitive, it plays a central role in casting supervised learning as a regularized factor model and later unification with unsupervised training principles that enables development of a principled semisupervised learning formulation in Chapter 6.

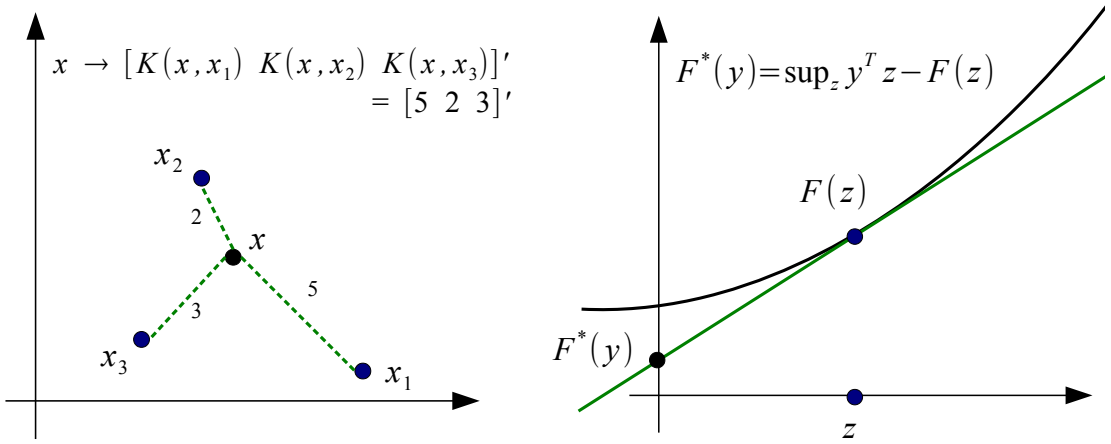
For reverse linear least squares, we seek an $n \times k$ matrix C that minimizes

$$\min_C \text{ tr}\left((X - CY)'(X - CY)\right). \tag{3.4}$$

This minimization can be easily solved to obtain the global solution $C = XY^\dagger = XY'(YY')^{-1}$. Interestingly, as long as X has full rank, n, the forward solution to (3.1) can be recovered from the solution of (3.4).[3] In particular, from the solutions of (3.1) and (3.4) we obtain the identity that $WXX' = YX' = YY'C'$, hence

$$W = YY'C'(XX')^{-1}. \tag{3.5}$$

[3] If X is not rank n, we can drop dependent columns.



(a) Kernel transformation

(b) Fenchel conjugate

Figure 3.1: (a) Transformed space using linear kernel. The new feature representation for $\mathbf{x} \in \mathbb{R}^2$ is a 3-dimensional vector that consists of the distances to the given samples, $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$. (b) The Fenchel conjugate, $F^\star$, which is the potential function for the reverse prediction problem. Intuitively, the value of the Fenchel conjugate at a point $\mathbf{y}$ is the intercept of the line with slope $\mathbf{y}$ that is tangent to F, which must be unique in this case by the strict convexity of F.

Once we add ridge regularization to the forward prediction problem, invertibility is assured and the forward solution can always be recovered from the reverse without any additional assumptions. For example, the optimal solution to the regularized problem (3.2) can always be recovered from the solution to the plain reverse problem (3.4) by using the straightforward identity that relates their solutions, $W(XX' + \alpha I) = YX' = YY'C'$, allowing one to conclude that

$$W = YY'C'(XX' + \alpha I)^{-1}.$$

Similarly, the kernelized reverse loss corresponds to

$$\min_B \operatorname{tr}\left((K - BY)'(K - BY)\right) \implies A = YY'B'(K + \alpha I)^{-1}K^{-1}$$

and the instance-weighted reverse loss is

$$\min_B \operatorname{tr}((K - BY)'(K - BY)\Lambda^2) \implies A = Y\Lambda Y'B'(K\Lambda + \alpha I)^{-1}.$$

I describe these derivations in Appendix B.2.

Therefore, for all the major variants of supervised least squares, one can solve the reverse problem and use the result to recover a solution to the forward problem.

3.1.3 Generalized forward-reverse connection

The connection between forward and reverse prediction can be extended beyond Gaussian distributions and least-squares losses to regular exponential family distribution and Bregman divergences.

Recall from 2.4 that maximizing likelihood of a regular exponential family corresponds to minimizing a regular Bregman divergence, $\max_{\boldsymbol{\theta}} p_F(\mathbf{z}|\boldsymbol{\theta}) \equiv \min_{\boldsymbol{\theta}} D_F(\boldsymbol{\theta} \;||\; \mathbf{f}^{-1}(\mathbf{z}))$. In this setting, $\boldsymbol{\theta} = WX_{:t}$ and $\mathbf{z} = Y_{:t}$, giving the generalized forward loss

$$\begin{aligned} L\left(WX\;;\;Y\right) &= \sum_{t=1}^{T} D_F(WX_{:t} \;||\; \mathbf{f}^{-1}(Y_{:t})) \\ &= \sum_{t=1}^{T} F(WX_{:t}) - F(\mathbf{f}^{-1}(Y_{:t})) - Y'_{:t}(WX_{:t} - \mathbf{f}^{-1}(Y_{:t})) \end{aligned} \tag{3.6}$$

The goal is to learn W such that $\mathbf{f}(WX_{:t}) \approx Y_{:t}$, using the forward prediction minimization

$$\min_W L\left(WX\;;\;Y\right) = \min_W \sum_{t=1}^{T} F(WX_{:t}) - Y'_{:t}WX_{:t}.$$

We derive the reverse loss using the notion of dual Bregman divergences; to do so, we need the following definitions. For a strictly convex potential $F : \mathbb{R}^n \to \mathbb{R}$, its Fenchel conjugate is

$$F^\star(\mathbf{y}) = \sup_{\mathbf{z}} \mathbf{z}'\mathbf{y} - F(\mathbf{z}).$$

Intuitively, as shown in Figure 3.1(b), the value of the Fenchel conjugate at a point $\mathbf{y}$ is the intercept of the line with slope $\mathbf{y}$ that is tanget to F, which must be unique in this case by the strict convexity of F. The associated transfer function, $\mathbf{f}^\star = \nabla F^\star$, satisfies $\mathbf{f}^\star = \mathbf{f}^{-1}$ (see proof of Lemma 2 in Appendix A.3).

The corresponding reverse loss for the transfer, $\mathbf{f}$, can therefore be written

$$R(X, CY) = D_{F^\star}(CY||\mathbf{f}(X)) = \sum_{t=1}^{T} F^\star(CY_{:t}) - F^\star(\mathbf{f}(X_{:t})) - X'_{:t}(CY_{:t} - \mathbf{f}(X_{:t})) \tag{3.7}$$

yielding the corresponding reverse loss minimization

$$\min_C \; R(X, CY) \equiv \min_C \sum_{t=1}^{T} F^\star(CY_{:t}) - X'_{:t}CY_{:t}. \tag{3.8}$$

As before, one can establish a unique correspondence between the optimal solutions to the forward training problem (3.6) and reverse training problem (3.8). In the general setting, there is no longer a clear closed-form mapping. Implicitly, however, such a one-to-one mapping still exists between the minimizers of the forward and reverse prediction problems: given C^*, one can uniquely recover W^* and vice versa.

Theorem 1 (Forward-Reverse Equivalence)**.** *Given an $n \times T$ input matrix X and a $k \times T$ output matrix Y, such that $T > \operatorname{rank}(X) = n$ and $T > \operatorname{rank}(Y) = k$, there exist unique global minimizers*

$$W^* = \arg\min_W D_F(WX||\mathbf{f}^{-1}(Y)) \tag{3.9}$$

$$C^* = \arg\min_C D_{F^\star}(CY||\mathbf{f}(X)) \tag{3.10}$$

$$\textit{where} \quad \mathbf{f}(W^*X)X' = YX' = Y\mathbf{f}^{-1}(C^*Y)'. \tag{3.11}$$

Proof Sketch: [Full proof in Appendix B.1]

The uniqueness of the minimizers results from the fact that F is strictly convex, making $G(\cdot) = F(\cdot X)$ strictly convex and so the optimization $\min_W G(W)$ therefore has a unique minimum.

For the correspondence, since W^* and C^* are global minimizers of $L(WX, Y)$ and $R(X, CY)$:

$$\begin{aligned}
&\frac{d}{dW} L(W^*X, Y) = (\mathbf{f}(W^*X) - Y)\, X' = \mathbf{0} \quad (k \times n) \\
&\frac{d}{dC} R(X, C^*Y) = (\mathbf{f}^\star(C^*Y) - X)\, Y' = \mathbf{0} \quad (n \times k) \\
\implies &\mathbf{f}(W^*X)X' = YX' \qquad \text{and} \qquad \mathbf{f}^\star(C^*Y)Y' = XY' \\
\implies &\mathbf{f}(W^*X)X' = Y\mathbf{f}^\star(C^*Y)'
\end{aligned}$$

■

As with least-squares losses, we can extend the optimization to include regularizers, kernels and instance weighting; these generalizations are described in Appendix B.2.

3.1.4 Case studies

In this section, I illustrate that two supervised learning algorithms, reduced rank regression and linear discriminant analysis, are related to unsupervised learning, specifically canonical correlation analysis. This connection is established using the reverse prediction formulation of supervised learning, i.e., using the factorization perspective of supervised learning. The goal of this section is to illustrate that viewing supervised learning as a regularized factor model elucidates interesting connections, further emphasized in the next section where many unsupervised learning problems can be represented as regularized factor models.

Reduced rank regression

For the above analysis, we have assumed that X and Y are full rank, with the assumption that dependent rows can be dropped to obtain linearly independent features and targets. In practice, however, it may be desirable to automatically select a subset of rows or reduce the dimension, particularly in cases where the data is high dimensional. One possible approach is to use a dimensionality reduction technique to first reduce the dimension of X and Y; however, this two-staged process means the choice of lower-dimensional manifold is not influenced by the regression problem.

Reduced rank regression[4] attempts to deal with this problem by restricting the rank of the weights in the regression: $\|Y - WX\|_F^2$ such that $\text{rank}(W) = k' < k$. To enforce this restriction, W is often factored into two matrices, $W = AB$ such that $A \in \mathbb{R}^{k \times k'}$ and $B \in \mathbb{R}^{k' \times n}$, giving

[4]Note that reduced rank regression is also called **simultaneous linear predictions** and **redundancy analysis** (van der Burg and de Leeuw, 1990).

the optimization

$$\min_{A,B} \|Y - ABX\|_F^2 = \min_{A,B:B'B=I} \|Y - ABX\|_F^2 \tag{3.12}$$

since we can rescale A to $AU\Sigma$ given singular value decomposition $B = U\Sigma V'$ and set $B = V'$, which satisfies $B'B = I$. The naive algorithm would be to alternate between minimizing A and B, which would result in local minima. Fortunately, however, there is a well-known solution to the reduced rank regression problem, where B is the eigenvectors for the generalized eigenvalue problem $BXY'Y'X = \Lambda BXX$ (van den Wollenberg, 1977).

Below, I show a new derivation that relates reduced rank regression to this generalized eigenvalue problem, using reverse prediction. By recognizing that BX is full-rank, we can equivalently consider the reverse prediction problem, $CY \approx BX$.

Proposition 1. *Reduced rank regression for Y such that $Y'(YY')^{-1}Y = Y'Y$ is a reverse prediction optimization,*

$$\operatorname*{argmin}_B \min_A \|Y - ABX\|_F^2 = \operatorname*{argmin}_{B:B'B=I} \min_C \|CY - BX\|_F^2 . \tag{3.13}$$

Moreover, this optimization gives the same solution as canonical correlation analysis.

Proof: The closed form solution for C is $C = BXY'(YY')^{-1}$. The optimization simplifies to

$$\min_{B:B'B=I} \left\|BXY'(YY')^{-1}Y - BX\right\|_F^2 = \min_{B:B'B=I} \left\|BX\left[Y^\dagger Y - I\right]\right\|_F^2 .$$

Since

$$\left[Y^\dagger Y - I\right]\left[Y^\dagger Y - I\right]' = Y^\dagger YY^\dagger Y - 2Y^\dagger Y + I = Y^\dagger Y - 2Y^\dagger Y + I = I - Y^\dagger Y$$

we obtain

$$\begin{aligned}
\min_{B:B'B=I} \left\|BX\left[Y^\dagger Y - I\right]\right\|_F^2 &= \min_{B:B'B=I} \operatorname{tr}(BX(I - Y^\dagger Y)X'B') \\
&= \min_{B:B'B=I} \operatorname{tr}(BXX'B') - \operatorname{tr}(BXY^\dagger YX'B') \\
&= \max_{B:B'B=I} \operatorname{tr}(BXY^\dagger YX'B') - \operatorname{tr}(BXX'B')
\end{aligned}$$

which is a well-known generalized eigenvalue problem, for eigenvalues Λ and eigenvectors B: $B(XY^\dagger YX') = \Lambda B(XX')$ (Kokiopoulou et al., 2011). Notice that $XY^\dagger YX'$ is actually several covariance matrices, i.e., $(XY')(YY)^{-1}(YX')$. So, B has the same solution it would for CCA, without requiring that XX' be invertible.

Now, since we assumed that Y is restricted to the set of targets that satisfy $Y'(YY)^{-1}Y = Y'Y$, we get that for these Y, B is the solution of the generalized eigenvalue problem $BXY'Y'X = \Lambda BXX$, which is the solution to reduced rank regression. ■

Linear discriminant analysis

Linear Discriminant Analysis (LDA) (Fisher, 1938) is a classification algorithm that computes a linear transformation of input data X maximizing the Euclidean distance between the means of the classes and minimizing the within-class variance for class indicator matrix $Y \in \{0,1\},\ \sum_j Y_{ij} = 1$). LDA has actually been shown to be an instance of reduced-rank regression (De la Torre, 2012), re-weighted by the number of samples in a class, $(YY')^{-1/2}$, giving

$$\min_{A,B} \left\| (YY')^{-1/2}(Y - ABX) \right\|_F^2$$

Notice that $\tilde{Y} = (YY')^{-1/2}Y$ satisfies the conditions of Proposition 1. Therefore, letting A absorb the re-weighting, $\tilde{A} = (YY')^{-1/2}A$, shows that LDA is equivalent to a reverse prediction optimization

$$\begin{aligned}
\operatorname*{argmin}_B \min_A \left\| (YY')^{-1/2}(Y - ABX) \right\|_F^2 &= \operatorname*{argmin}_B \min_A \left\| \tilde{Y} - (YY')^{-1/2}ABX \right\|_F^2 \\
&= \operatorname*{argmin}_B \min_{\tilde{A}} \left\| \tilde{Y} - \tilde{A}BX \right\|_F^2 \\
&= \operatorname*{argmin}_{B:B'B=I} \min_C \|CY - BX\|_F^2
\end{aligned}$$

and so is equivalent to canonical correlation analysis. LDA has also previously been shown to be an instance of CCA using a different analysis than the one above (Bach and Jordan, 2006).

3.2 Unsupervised learning as factor models

Unsupervised learning is a foundational problem in machine learning and statistics, encompassing problems as diverse as clustering, dimensionality reduction, system identification, and grammar induction. Unlike supervised learning, the goal in unsupervised learning is to learn with *unlabeled* data, i.e., with no given targets. For example, classical methods such as principal components analysis and k-means clustering are derived from principles for re-representing the input data, rather than minimizing prediction error on any associated output variables.

In this section, we see that a surprising number of unsupervised learning problems are actually regularized factor models. These connections have been noticed several times; I discuss these pairwise connections and unification frameworks in the next section. In this section, to develop a sense for these connections, I give three case studies unified under regularized factor models in my own work: generalized kernel principal components analysis (White and Schuurmans, 2012), canonical correlation analysis (White et al., 2012) and normalized cut (Xu et al., 2009). I provide a summary table of the remaining unsupervised learning algorithms that can be viewed as a regularized factor-

ization, to the best of my knowledge; more details on each is given in the Appendix C, including references to the papers that originally proved the connection.

3.2.1 Forward-reverse prediction for unsupervised learning

Unsupervised learning can be framed similarly to supervised learning, but with an outer minimization that must also impute the unknown targets (labels). Using the forward loss, however, is problematic; to see why, consider the forward loss

$$\min_{W \in \mathbb{R}^{k \times n}, \Phi \in \mathbb{R}^{k \times T}} \|WX - \Phi\|_F^2$$

where now Φ are the unknown targets, since we are not given Y. As before, each k-dimensional column corresponds to a sample. Unlike supervised learning, however, forward prediction for unsupervised learning is vacuous. For any given W, Φ can be set to $\Phi = WX$, giving the optimal error of zero. Thus, the standard forward view has no ability to distinguish between alternative parameter matrices W.

Similarly, for prediction with Bregman divergences, the forward minimization

$$\min_{W,\Phi} D_F(WX \,||\, \mathbf{f}^{-1}(\Phi)) = 0$$

since for any W, we can set $\Phi = \mathbf{f}(WX)$. Therefore, unlike supervised learning which can be formulated as either forward or reverse prediction, unsupervised learning must be framed in terms of reverse prediction, i.e., as regularized factor models.

3.2.2 Case studies

In this section I present three derivations showing that three varied unsupervised learning algorithms can be viewed as regularized factorization. A summary table is included in Section 3.4 for the many other unsupervised learning algorithms that can be so formulated, with more detailed information about each in Appendix C.

Exponential family PCA

For intuition, I first show this result for standard principal components analysis (PCA).

Proposition 2. ***Principal components analysis*** *solves a regularized factor model with a least-squares loss and no constraints or regularizers*

$$\min_{C \in \mathbb{R}^{n \times k}} \min_{\Phi \in \mathbb{R}^{k \times T}} \|X - C\Phi\|_F^2 = \min_{C \in \mathbb{R}^{n \times k}} \min_{\Phi \in \mathbb{R}^{k \times T}} tr\left((X - C\Phi)'(X - C\Phi)\right) \tag{3.14}$$

Proof: Because we have a closed form solution $C = X\Phi^\dagger$, from the solution of (3.4), we get

$$\begin{aligned}\operatorname*{argmin}_{\Phi\in\mathbb{R}^{k\times T}} \min_{C\in\mathbb{R}^{n\times k}} \|X - C\Phi\|_F^2 &= \operatorname*{argmin}_{\Phi\in\mathbb{R}^{k\times T}} \operatorname{tr}\left((I-\Phi^\dagger\Phi)'X'X(I-\Phi^\dagger\Phi)\right) \\ &= \operatorname*{argmin}_{\Phi\in\mathbb{R}^{k\times T}} \operatorname{tr}\left((I-\Phi^\dagger\Phi)X'X\right) \qquad (3.15)\end{aligned}$$

where the second equivalence follows from the fact that $\Phi^\dagger\Phi$ is symmetric and $(I - \Phi^\dagger\Phi)(I - \Phi^\dagger\Phi)' = I - 2\Phi^\dagger\Phi + \Phi^\dagger\Phi\Phi^\dagger\Phi = I - \Phi^\dagger\Phi$. Now we can obtain an equivalent maximization

$$(3.15) = \operatorname*{argmin}_{\Phi\in\mathbb{R}^{k\times T}} \operatorname{tr}\left(X'X\right) - \operatorname{tr}\left(\Phi^\dagger\Phi X'X\right) = \operatorname*{argmax}_{\Phi\in\mathbb{R}^{k\times T}} \operatorname{tr}\left(\Phi^\dagger\Phi X'X\right).$$

Using the singular value decomposition of Φ, $\Phi = U\Sigma V'$ for $U'U = I$, $V'V = I$ and Σ diagonal, then

$$\begin{aligned}&\Phi^\dagger = \Phi'(\Phi\Phi')^{-1} = V\Sigma U'(U\Sigma V'V\Sigma U')^{-1} = V\Sigma U'(U\Sigma^2 U')^{-1} = V\Sigma U'U\Sigma^{-2}U' = V\Sigma^{-1}U' \\ &\implies \Phi^\dagger\Phi = V\Sigma^{-1}U'U\Sigma V' = VV'\end{aligned}$$

allowing the optimization variable to be simplified with an added constraint

$$\max_{\Phi\in\mathbb{R}^{k\times T}} \operatorname{tr}\left(\Phi^\dagger\Phi X'X\right) \quad = \quad \max_{\Phi\in\mathbb{R}^{k\times T}:\Phi'\Phi=I} \operatorname{tr}\left(\Phi'\Phi X'X\right).$$

The well-known solution to this maximization is given by the top k eigenvectors of $X'X$ (Overton and Womersley, 1993), which is equivalent to the right singular vectors of $X = U\Sigma V'$. Correspondingly, $C = X\Phi^\dagger = U\Sigma V'V' = U\Sigma$. ■

This same observation has been made in the statistics literature (Jong and Kotz, 1999); we extend it also to kernel PCA.

Proposition 3. ***Kernel principal components analysis*** *solves a regularized factor model for a kernelized least-squares loss and no constraints or regularizers*

$$\min_{B\in\mathbb{R}^{T\times k}} \min_{\Phi\in\mathbb{R}^{k\times T}} \|K - B\Phi\|_F^2 \qquad (3.16)$$

Proof: From (Schölkopf et al., 1997), kernel PCA consists of finding the eigenvectors of K. We know that the solution to $\min_{B\in\mathbb{R}^{T\times k}} \min_{\Phi\in\mathbb{R}^{k\times T}} \|K - B\Phi\|_F^2$ is the top k eigenvectors of $K'K = K^2$ from Proposition 2. Moreover, K^2 has the same eigenvectors as K since $K = Q\Lambda Q'$ gives $K^2 = Q\Lambda^2 Q'$. ■

Exponential family PCA generalizes PCA from a Gaussian assumption on the data to any exponentially family distribution (Collins et al., 2002). Originally, exponential family PCA was introduced as the optimization

$$\min_C \min_\phi -\log(p_F(\mathbf{x}|C\phi)) \equiv \min_C \min_\phi D_F(\mathbf{x}||\mathbf{f}^{-1}(C\phi)) \qquad (3.17)$$

where $C\phi$ is the parameter for the distribution, which corresponds to the projection into the lower-dimensional subspace, since $C \in \mathbb{R}^{n\times k}$ has $k < n$.

Notice that the optimization on $D_F(\mathbf{x}||\mathbf{f}^{-1}(\boldsymbol{\theta}))$ is not necessarily convex, as Bregman divergences are only guaranteed to be convex in the first argument (see Section 2.2 for more on Bregman divergences). In the following, I show that exponential family PCA is an instance of regularized factorization, framed now as a convex optimization.

Lemma 2. *With* $\hat{\mathbf{y}} = \mathbf{f}(\hat{\mathbf{z}})$ *and* $\mathbf{y} = \mathbf{f}(\mathbf{z})$,

$$D_F(\hat{\mathbf{z}}||\mathbf{z}) = D_{F^\star}(\mathbf{y}||\hat{\mathbf{y}}). \tag{3.18}$$

Proof Sketch: [Full proof in Appendix A.3]
The result mainly follows from the connection between $F^\star$ and F. Since $F^\star(\mathbf{y}) = \max_{\mathbf{z}} \mathbf{z}'\mathbf{y} - F(\mathbf{z})$, we can solve for the maximum $\mathbf{z}$ to obtain

$$\frac{d}{d\mathbf{z}} = \mathbf{y} - \nabla F(\mathbf{z}) = \mathbf{y} - \mathbf{f}(\mathbf{z}) = 0 \implies \mathbf{z} = \mathbf{f}^{-1}(\mathbf{y})$$

giving

$$F^\star(\mathbf{y}) = \mathbf{f}^{-1}(\mathbf{y})'\mathbf{y} - F(\mathbf{f}^{-1}(\mathbf{y}))$$

The remainder of the proof uses this equivalence to rewrite $D_{F^\star}$ in terms of F and $\mathbf{f}^{-1} = \mathbf{f}^\star$. ■

Proposition 4. ***Exponential family PCA*** *solves a regularized factor model with a regular Bregman divergence loss, $D_{F^\star}$ (corresponding to a regular exponential family distribution), and no constraints or regularizers*

$$\min_{C\in\mathbb{R}^{n\times k}} \min_{\Phi\in\mathbb{R}^{k\times T}} D_{F^\star}(C\Phi||\mathbf{f}(X)) \tag{3.19}$$

Proof: From Banerjee et al. (2005) and the summarized discussion in Section 2.2, we know that minimizing the log likelihood for regular exponential families corresponds to minimizing a regular Bregman divergence, giving the minimization in (3.17). This is not a convex minimization, but using Lemma 2, we convert the minimization to a dual space,

$$\begin{aligned} D_{F^\star}(C\Phi||\mathbf{f}(X)) &= D_F(\mathbf{f}^{-1}(\mathbf{f}(X))||\mathbf{f}^{-1}(C\Phi)) \\ &= D_F(X||\mathbf{f}^{-1}(C\Phi)) \end{aligned}$$

because $(F^\star)^\star = F$. Therefore, (3.19) is equivalent to the loss solved by exponential family PCA. ■

Moreover, based on the kernel extensions for generalized reverse prediction given in Appendix B.2, one can extend exponential family PCA to *kernel* exponential family PCA.

Proposition 5. ***Kernel exponential family PCA*** *solves a regularized factor model with regular Bregman divergence, $D_{F^\star}$, and no constraints or regularizers*

$$\min_{\Phi \in \mathbb{R}^{k \times T}} \min_{B \in \mathbb{R}^{T \times k}} D_{F^\star}(C\Phi || \mathbf{f}(K)) \tag{3.20}$$

This optimization reduces to kernel PCA in (3.16) for a Gaussian exponential family distribution.

Proof: The kernelized reverse optimization follows from Appendix B.2. For completeness, I show the second statement, even though it follows simply from the fact that a Gaussian exponential family corresponds to a least-squares Bregman divergence. The potential (or cumulant) for a Gaussian distribution is $F^\star(\mathbf{x}) = \frac{1}{2}\mathbf{x}'\mathbf{x}$, giving $\nabla F^\star(\mathbf{x}) = \mathbf{x}$, where $\mathbf{f}^\star(\mathbf{x}) = \nabla F^\star(\mathbf{x})$. Recall that $\mathbf{f}^\star = \mathbf{f}^{-1}$ and so $\mathbf{f}(\mathbf{x}) = \mathbf{x}$; therefore, we obtain

$$\begin{aligned} D_{F^\star}(C\boldsymbol{\phi} \,||\, \mathbf{k}) &= F^\star(C\boldsymbol{\phi}) - F^\star(\mathbf{f}(\mathbf{k})) - \mathbf{f}^\star(\mathbf{f}(\mathbf{k}))'(C\boldsymbol{\phi} - \mathbf{k}) \\ &= \frac{1}{2}\boldsymbol{\phi}'C'C\boldsymbol{\phi} - \frac{1}{2}\mathbf{k}'\mathbf{k} - \mathbf{k}'(C\boldsymbol{\phi}) + \mathbf{k}'\mathbf{k} \\ &= \frac{1}{2}\boldsymbol{\phi}'C'C\boldsymbol{\phi} - \mathbf{k}'(C\boldsymbol{\phi}) + \frac{1}{2}\mathbf{k}'\mathbf{k} = \frac{1}{2}\left\|C\boldsymbol{\phi} - \mathbf{k}\right\|_2 \end{aligned}$$

giving

$$\min_{C,\Phi} D_{F^\star}(C\Phi \,||\, K) = \min_{C,\Phi} \sum_{i=1}^{T} D_{F^\star}(C\Phi_{:i} \,||\, K_{:i}) = \min_{C,\Phi} \frac{1}{2}\sum_{i=1}^{T} \|C\Phi - K\|_F^2 .$$

From Proposition 2, the solution to this optimization is the top k eigenvectors of $K'K = K^2$, which are the same as the eigenvectors for K, since $K = Q\Lambda Q'$ giving $K^2 = Q\Lambda^2 Q'$. ■

Canonical correlation analysis

Canonical correlation analysis (CCA) (Hotelling, 1936), is typically expressed as the problem of projecting two views X_1 and X_2 so that the correlation between them is maximized (Hardoon et al., 2004): $\max_{u,v} \operatorname{corr}(u'X_1, v'X_2)$. Assuming the data is centered (i.e. $X_1\mathbf{1} = \mathbf{0}$ and $X_2\mathbf{1} = \mathbf{0}$), the sample covariance of X_1 and X_2 is given by $X_1{X_1}'/T$ and $X_2{X_2}'/T$ respectively. CCA can then be expressed as an optimization over matrix variables $U \in \mathbb{R}^{n \times k}$ and $V \in \mathbb{R}^{m \times k}$ (De Bie et al., 2005),

$$\max_{U,V} \operatorname{tr}\left(U'X_1X_2'V\right) \;\; s.t. \;\; U'X_1{X_1}'U = V'X_2{X_2}'V = I. \tag{3.21}$$

The closed form solution for U is the top k eigenvectors of

$$\left(X_1{X_1}'\right)^{-1}\left(X_1{X_2}'\right)\left(X_2{X_2}'\right)^{-1}\left(X_2{X_1}'\right)$$

and the canonical variables (or directions) are $U'X_1$.

Although not clear from this classical formulation (3.21), CCA can be expressed by a generative model: given a latent representation, $\boldsymbol{\phi}_j$, the observations $\mathbf{x}_j = C^{(1)}\boldsymbol{\phi}_j + \boldsymbol{\epsilon}_j$ and $\mathbf{y}_j = C^{(2)}\boldsymbol{\phi}_j + \boldsymbol{\nu}_j$ are generated by a linear mapping plus independent zero mean Gaussian noise, $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \Sigma_x)$, $\boldsymbol{\nu} \sim N(\mathbf{0}, \Sigma_y)$ (Bach and Jordan, 2006). We further clarified this connection to regularized factorization (White et al., 2012); to the best of my knowledge, this formulation of CCA given in Proposition 6 had not previously been shown [5].

Proposition 6. ***Canonical correlation analysis*** *solves a regularized factor model with no constraints or regularizer and least squares loss with normalized data matrices,* $\tilde{Z}=\begin{bmatrix} (X_1X_1')^{-\frac{1}{2}}X_1 \\ (X_2X_2')^{-\frac{1}{2}}X_2 \end{bmatrix}$

$$(C, \Phi) = \underset{C\in\mathbb{R}^{(n_1+n_2)\times k}, \Phi\in\mathbb{R}^{k\times T}}{\operatorname{argmin}} \|\tilde{Z} - C\Phi\|_F^2 \tag{3.22}$$

where $C = \begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix}$. *In terms of classical CCA,* $U = (X_1X_1')^{-\frac{1}{2}} C^{(1)}$ *and* $V = (X_2X_2')^{-\frac{1}{2}} C^{(2)}$ *provide an optimal solution to (3.21), implying* $C^{(1)\prime}C^{(1)} = C^{(2)\prime}C^{(2)} = I$ *is satisfied in the solution to (3.22).*

Proof: To show that (3.21) and (3.22) have equivalent solutions we exploit some developments from Sun et al. (2011). Let $N = (X_1X_1')^{-\frac{1}{2}}$ and $M = (X_2X_2')^{-\frac{1}{2}}$, hence

$$\tilde{Z}\tilde{Z}' = \begin{bmatrix} I & NX_1X_2'M \\ MX_2X_1'N & I \end{bmatrix}.$$

First consider (3.21). Its solution can be characterized by the maximal solutions to the generalized eigenvalue problem (De Bie et al., 2005):

$$\begin{bmatrix} 0 & X_1X_2' \\ X_2X_1' & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} X_1X_1' & 0 \\ 0 & X_2X_2' \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix},$$

which, under the change of variables $\mathbf{u} = N\mathbf{a}$ and $\mathbf{v} = M\mathbf{b}$, is equivalent to

$$\begin{aligned}
\equiv \quad & \begin{bmatrix} 0 & X_1X_2'M \\ X_2X_1'N & 0 \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \lambda \begin{bmatrix} N^{-1} & 0 \\ 0 & M^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \\
\equiv \quad & \begin{bmatrix} 0 & NX_1X_2'M \\ MX_2X_1'N & 0 \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \lambda \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \\
\equiv \quad & \tilde{Z}\tilde{Z}' \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = (\lambda + 1) \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}
\end{aligned}$$

where shifting the eigenvalues by 1 from adding an identity matrix to both sides on the last line did not change the eigenvectors. By setting $\begin{bmatrix} A \\ B \end{bmatrix}$ to the top k eigenvectors of $\tilde{Z}\tilde{Z}'$ one can show that $U = NA$ and $V = MB$ provides an optimal solution to (3.21) (De Bie et al., 2005).

[5] (Long et al., 2008) gave a similar but not equivalent formulation to (3.22), due to the lack of normalization.

By comparison, for (3.22), an optimal Φ is given by $\Phi = C^\dagger \tilde{Z}$, where $C^\dagger$ denotes pseudo-inverse. Hence

$$\operatorname*{argmin}_C \min_\Phi \|\tilde{Z} - C\Phi\|_F^2 = \operatorname*{argmin}_C \|(I - CC^\dagger)\tilde{Z}\|_F^2.$$

From the same analysis as in Proposition 2, we know that this optimization corresponds to

$$\max_{C:C'C=I} \operatorname{tr}(CC'\tilde{Z}\tilde{Z}').$$

where the solution is given by the top k eigenvectors of $\tilde{Z}\tilde{Z}'$ (Overton and Womersley, 1993), i.e., the left singular vectors of $\tilde{Z} = U\Sigma V'$. Correspondingly, $\Phi = C^\dagger \tilde{Z} = U'U\Sigma V' = \Sigma V'$. ■

Corollary 1. *The joint generalized eigenvalue problem* $S_1X_1X_2'S_2'C^{(1)} = C^{(1)}\Lambda$ *and* $S_2X_2X_1'S_1'C^{(2)} = C^{(2)}\Lambda$*, for* S_1 *and* S_2 *invertible and* $S_1X_1X_2'S_2'$ *diagonalizable, is a regularized factor model:*

$$(C, \Phi) = \operatorname*{argmin}_{C \in \mathbb{R}^{(n_1+n_2)\times k}, \Phi \in \mathbb{R}^{k\times T}} \left\| \begin{bmatrix} S_1X_1 \\ S_2X_2 \end{bmatrix} - C\Phi \right\|_F^2 \tag{3.23}$$

If $X_1 = X_2$ *and* $S_1 = S_2$*, this reduces to one eigenvalue decomposition,* $S_1X_1X_1'S_1'C^{(1)} = C^{(1)}\Lambda$.

There are several advantages to having a regularized factorization view of CCA. First, as we will see in Section 5.3, CCA is the classical formulation of two-view subspace learning; this connection, therefore, enables clear extensions to two-view learning by using different modeling choices in the regularized factor model. Second, reduced-rank regression algorithms can be cast as CCA with the input data as one view and the output data (targets) as the other view, as described in Section 3.1.4. This combined goal of unsupervised, dimensionality reduction on one view (the input data) and supervised regression on the other view will also be useful for formulating semi-supervised learning convexly in Chapter 6. Finally, for generalizing the loss, there is a distinct advantage to the formulation in (3.22) over the generalized eigenvalue formulation or the one introduced for reduced rank regression in Section 3.1.4. The CCA formulation in (3.22) naturally enables the least-squares loss to be generalized to any Bregman divergence, $D_{F^\star}(C\Phi||\mathbf{f}(\tilde{Z}))$, where convexity is maintained since a Bregman divergence is guaranteed to be convex in the first argument. This extension is not obvious for generalized eigenvalue problems or for Equation (3.13), since $D_{F^\star}(CX_2||\mathbf{f}(BX_1))$ is not convex in terms of B.

Normalized cut (spectral clustering)

Spectral clustering algorithms, such as normalized cut, were introduced with the intuition that the eigenvectors of an adjacency graph could enable points to be partitioned (Dhillon et al., 2004). Below, we show that normalized cut solves a regularized factor model. As far as I know, this connection has not been previously realized. These results simplify some of the connections observed

in (Dhillon et al., 2004; Chen and Peng, 2008; Kulis et al., 2009) relating k-means to normalized cut, but generalizes them to relate to supervised least squares and regularized factor models.

Proposition 7. ***Normalized cut*** *solves a regularized factor model with a constraint set on* Φ, $\mathcal{F} = \{\Phi \in \{0,1\}^{k\times T}, \mathbf{1}\Phi = \mathbf{1}\}$, *no constraints on* C, *no regularizers and an instance-weighted normalized kernel least squares loss for kernel* $K \geq 0$ *and normalizer* $\Lambda = \operatorname{diag}(\mathbf{1}K)$*:*

$$\min_{\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \min_{C} \left\|(K\Lambda^{-2} - C\Phi)\Lambda\right\|_F^2 \tag{3.24}$$

Proof: Let $\tilde{K} = K\Lambda^{-2}$ be the normalized kernel, then

$$\begin{aligned}
\left\|(K\Lambda^{-2} - C\Phi)\Lambda\right\|_F^2 &= \operatorname{tr}\left(\Lambda(\tilde{K} - C\Phi)'(\tilde{K} - C\Phi)\Lambda\right) \\
&= \operatorname{tr}(\Lambda\tilde{K}\tilde{K}\Lambda) - 2\operatorname{tr}(\Lambda\tilde{K}C\Phi\Lambda) + \operatorname{tr}(\Phi\Lambda^2\tilde{K}C) \\
\implies & -\tilde{K}\Lambda^2\Phi' + C\Phi\Lambda^2\Phi' = \mathbf{0} \\
\implies & C = \tilde{K}\Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}
\end{aligned}$$

We can plug this solution back in and obtain

$$\operatorname{tr}\left((I - \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi)K^2(I - \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi)\Lambda^2\right).$$

Now we can further simplify

$$\begin{aligned}
&(I - \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi)\Lambda^2(I - \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi)' \\
&\quad = \Lambda^2 - 2\Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2 + \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2 \\
&\quad = \Lambda^2 - 2\Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2 + \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2 \\
&\quad = \Lambda^2 - \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2
\end{aligned}$$

giving

$$\operatorname{tr}\left((I - \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi)\tilde{K}^2(I - \Lambda^2\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi)\Lambda^2\right) = \operatorname{tr}\left((I - \Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2)\tilde{K}^2\Lambda^2\right)$$

and simplifying the optimization to

$$\begin{aligned}
\operatorname*{argmin}_{\Phi} \min_{C} \left\|(K\Lambda^{-2} - C\Phi)\Lambda\right\|_F^2 &= \operatorname*{argmin}_{\Phi} \operatorname{tr}\left((I - \Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2)\tilde{K}^2\Lambda^2\right) \\
&= \operatorname*{argmin}_{\Phi} \operatorname{tr}(\tilde{K}^2\Lambda^2) - \operatorname{tr}\left(\Phi'(\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2\tilde{K}^2\Lambda^2\right) \\
&= \operatorname*{argmax}_{\Phi} \operatorname{tr}\left((\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda^2\tilde{K}^2\Lambda^2\Phi'\right) \\
&= \operatorname*{argmax}_{\Phi} \operatorname{tr}\left((\Phi\Lambda^2\Phi')^{-1}\Phi\Lambda(\Lambda^{-1}K^2\Lambda^{-1})\Lambda\Phi'\right).
\end{aligned}$$

Finally, equivalently, we can solve for $\tilde{\Phi} = \Phi\Lambda$, giving

$$\operatorname*{argmax}_{\tilde{\Phi}} \operatorname{tr}\left((\tilde{\Phi}\tilde{\Phi}')^{-1}\tilde{\Phi}(\Lambda^{-1}K^2\Lambda^{-1})\tilde{\Phi}'\right)$$

where the solution is the top k eigenvectors of $\Lambda^{-1}K^2\Lambda^{-1}$ and so equivalently of $\sqrt{\Lambda^{-1}K^2\Lambda^{-1}} = \Lambda^{-1/2}K\Lambda^{-1/2}$, which is the same as the solution to normalized cut (Dhillon et al., 2004).

For completeness, I also show the original way that this connection was illustrated (Xu et al., 2009). Using the standard transformation $C = XB$, used in Appendix B.2.1 to indicate that for the least-squares setting we can factor out one of the kernel matrices to obtain a simplified reverse loss, we obtain the loss:

$$\min_{\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \min_{B} \operatorname{tr}\left((X\Lambda^{-1} - XB\Phi)'(X\Lambda^{-1} - XB\Phi)\Lambda\right) \tag{3.25}$$

$$= \min_{\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \min_{B} \operatorname{tr}\left((\Lambda^{-1} - B\Phi)'K(\Lambda^{-1} - B\Phi)\Lambda\right) \tag{3.26}$$

For any Φ, the inner minimization can be solved to obtain $B = \Phi'(\Phi\Lambda\Phi')^{-1}$. Substituting back into the objective and reducing yields

$$\min_{\Phi:\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \operatorname{tr}\left((\Lambda^{-1} - \Phi'(\Phi\Lambda\Phi')^{-1}\Phi)'K(\Lambda^{-1} - \Phi'(\Phi\Lambda\Phi')^{-1}\Phi)\Lambda\right) \tag{3.27}$$

$$= \min_{\Phi:\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \operatorname{tr}\left((\Lambda^{-1} - \Phi'(\Phi\Lambda\Phi')^{-1}\Phi)K\right) \tag{3.28}$$

The first term is constant and can be altered without affecting the minimizer, hence (3.28) is equivalent to

$$\min_{\Phi:\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \operatorname{tr}(I) - \operatorname{tr}\left(\Phi'(\Phi\Lambda\Phi')^{-1}\Phi K\right) \tag{3.29}$$

$$= \min_{\Phi:\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \operatorname{tr}\left((\Phi\Lambda\Phi')^{-1}\Phi(\Lambda - K)\Phi'\right) \tag{3.30}$$

Xing and Jordan (2003) have shown that with $\Lambda = \operatorname{diag}(\mathbf{1}K)$, $\Lambda - K$ is the Laplacian and (3.30) is equivalent to normalized cut. ■

These simple connections are important both algorithmically and theoretically. For example, see (Dhillon et al., 2004) for a discussion on using spectral clustering to initialize k-means and using simpler k-means algorithms to avoid expensive eigenvector computations for spectral clustering. Moreover, we can now extend normalized cut to general Bregman divergences with instance weights $\Lambda = \operatorname{diag}(\mathbf{1}K)$. Weighting with the kernel matrix intuitively places the most representative points as cluster centers, since the error is high for classifying those points incorrectly; the sum of edge weights across cuts in the adjacency graph, therefore, should be correspondingly low. This extension indicates how a unified formalism is not only powerful for relating known algorithms but also for producing novel algorithms. The unified framework makes algorithmic gaps clear; without the unification, the idea of a generalized version of normalized-cut was not clear. See Appendix C.4.2 for more on this generalization.

3.3 Previous frameworks and unifications

There have been several investigations into the relationships between statistical machine learning algorithms, particularly through eigenvalue decomposition and under least-squares optimizations. I

will first describe these simpler frameworks and then discuss generalized frameworks.

Eigenvalue-decomposition frameworks unify algorithms as instances of a generalized eigenvalue decomposition. Borga et al. (1997) showed that PCA, partial least squares, CCA and reduced-rank regression are all instances of a generalized eigenproblem, with different focuses on variance, covariance and correlation. Weiss (1999) unified three image segmentation algorithms, including normalized cut, as eigenvalue problems. Yan et al. (2007) unified dimensionality reduction techniques, including PCA, locality preserving projections, Isomap, and linear discriminant analysis, under a graph embedding formalism that reduces to a generalized eigenvalue computation. Kokiopoulou et al. (2011) provided an extensive unification of dimensionality reduction techniques as constrained trace maximizations, corresponding to generalized eigenvalue problems, including Laplacian eigenmaps, orthogonal neighbourhood preserving projections, Isomap and metric multidimensionsal scaling. Liski et al. (2013) introduced a formalism called supervised invariant coordinate selection that again consists of a generalized eigendecomposition, unifying linear discriminant analysis, canonical correlation analysis, sliced inverse regression, sliced average variance estimate, directional regression, and principal Hessian directions. From Corollary 1, many of these algorithms correspondingly solve a regularized factor model, since generalized eigenvalue problems can be written as a regularized factorization;[6] if a generalized eigenvalue problem satisfies the conditions of Corollary 1, it is included in the Summary table in Section 3.4.

Least-squares frameworks, like the regularized factor models formalism, attempt to unify algorithms under a least-squares loss with different settings, such as different kernels and instance weighting. De la Torre (2012) related several component analysis techniques, including PCA, linear discriminant analysis, CCA, Laplacian eigenmaps, locality preserving projections, local linear embeddings, neighbourhood preserving embeddings, and normalized cut as particular instances of least-squares weighted kernel reduced rank regression. This framework is similar to least-squares regularized factor model, with a focus, however, of regressing onto output targets. Roweis and Ghahramani (1999) unified probabilistic PCA, mixture-model clustering, vector quantization, Kalman filter models and hidden Markov models under *linear dynamical systems*. A linear dynamical system is a factored system, but with a temporal connection between the latent variables:

$$\Phi_t = A\Phi_{t-1} + \eta_t$$
$$X_t = C\Phi_t + \nu_t$$

for zero-mean noise variables $\eta_t \sim \mathcal{N}(0, Q)$ and $\nu_t \sim \mathcal{N}(0, R)$ with covariance matrices Q and R. For the non-temporal setting, such as probabilistic PCA, they simply remove the temporal structure

[6] This connection to least-squares is also noted by Sun et al. (2009a), for a smaller set than in Corollary 1

by setting $A = 0$ and so making $\Phi_t = \eta_t$. The resulting formulation is very similar to a regularized factor model, but with the variance of η_t explicitly learned. Though this temporal factored analysis is more general, and may be an interesting direction for unification in the future, the basic, non-temporal factorization merits further understanding before moving on to this more complicated formalism. Moreover, this work did not provide new algorithms, whereas several algorithms have been developed while considering the simpler class of regularized factor models Singh and Gordon (2008).

Generalizations to Bregman divergences have also been explored as unifying formalisms, relaxing the least-squares (Gaussian) assumption to enable more general losses (distributions). Gordon (2003) introduced generalized^2 linear^2 models ((GL)^{2}M), where for three link functions $\mathbf{f}$, $\mathbf{g}$, and $\mathbf{h}$, the goal is to learn C and Φ such that $X = \mathbf{f}(\mathbf{g}(C)\mathbf{h}(\Phi))$. This formalism generalizes beyond regularized factor models, enabling further unification including independent component analysis. The optimization of (GL)^{2}Ms, however, is much more difficult. Singh and Gordon (2008) moved away from this more general formulation and proposed almost exactly the same regularized factor models optimization, but with a focus on weighted Bregman divergences instead of general losses. Under this formalism, they unified several of the algorithms listed in the summary table in Section 3.4 and, importantly, provided new optimization algorithms for these problems with a useful generic Newton-update algorithm. Banerjee et al. (2005) discuss the unification of several clustering algorithms under Bregman information, connecting these approaches to rate distortion theory. Though similar, this approach uses general Bregman divergences to generalize properties of the optimization but to learn the linear factorization $X \approx C\Phi$. Buntine and Jakulin (2006) unifies several algorithms for discrete data as discrete component analysis, including discrete independent component analysis, latent Dirichlet allocation and probabilistic latent semantic indexing.

Pairwise connections have also previously been observed, without a focus on a unifying formalism. Some of these include the connection between PCA and k-means (Jong and Kotz, 1999); k-means and normalized cut (Dhillon et al., 2004; Chen and Peng, 2008; Kulis et al., 2009); CCA and LDA (Bach and Jordan, 2006); correspondence analysis and latent class analysis (van der Heijden et al., 1999); and non-negative matrix factorization and probabilistic latent semantic analysis (Gaussier and Goutte, 2005).

These unifications facilitate a broader understanding of the properties of a wide range of learning techniques. In addition to elucidating connections between algorithms, it also highlights the utility in separating the problem from the algorithm. By focusing on this unified optimization with different modeling choices, we will see useful advances in generic convex reformulation techniques in the next chapter that enable important advances in statistical machine learning problems.

3.4 Summary table

If the entry is empty, this specifies no regularization and no constraints. The hard clustering set is $\mathcal{C}_{\text{hard}} = \{\Phi \in \{0,1\}^{k \times T}, \mathbf{1}\Phi = \mathbf{1}\}$.

ALGORITHM	LOSS	CONSTRAINTS & REGULARIZERS
CCA (PROP. 6) $\equiv$ ORTHONORMAL PLS	$\left\| \begin{bmatrix} (X_1 X_1')^{-1} X_1 \\ (X_2 X_2')^{-1} X_2 \end{bmatrix} - C\Phi \right\|_F^2$	
INFORMATION THEORETIC CLUSTERING (APP. C.4.4)	$D_F(\mathbf{x} \,\|\| \, C\phi)$ WITH $F(\mathbf{x}) = \sum_{i=1}^n \mathbf{x}(i) \log \mathbf{x}(i) - \mathbf{1}'\mathbf{x}$	$\mathcal{C}_{\text{HARD}}$
ISOMAP (APP. C.2.1)	$\|K - C\Phi\|_F^2$ $K = -\frac{1}{2}(I - \mathbf{e}\mathbf{e}')S(I - \mathbf{e}\mathbf{e}')$ WITH $S_{i,j} = \|X_{:i} - X_{:j}\|$	
K-MEANS CLUSTERING (APP. C.3, PROP. 11)	$\|X - C\Phi\|_F^2$	$\mathcal{C}_{\text{HARD}}$
K-MEDIANS CLUSTERING (SEE SINGH AND GORDON 2008)	$\|X - C\Phi\|_{1,1}$	$\mathcal{C}_{\text{HARD}}$
LAPLACIAN EIGENMAPS $\equiv$ KERNEL LPP (APP. C.2.2)	$\|K - C\Phi\|_F^2$ FOR $K = L^\dagger$	
LINDE-BUZO-GRAY (APP. C.4.3)	$D_F(\mathbf{x}\|\|C\phi) = \frac{\mathbf{x}}{C\phi} - \log \frac{\mathbf{x}}{C\phi} - 1$ WITH $F = -\log \mathbf{x}$	$\mathcal{C}_{\text{HARD}}$
LOCALLY LINEAR EMBEDDINGS (APP. C.2.3)	$\|K - C\Phi\|_F^2$ $K = (I - \mathbf{e}\mathbf{e}')\tilde{K}(1 - \mathbf{e}\mathbf{e}')$ WITH $\tilde{K} = (\lambda_{\text{MAX}}(M)I - M)$	
MAXIMUM MARGIN MATRIX FACTORIZATION (SEE SINGH AND GORDON 2008)	$\sum_{r=1}^{R-1} \sum_{ij:X_{ij} \neq 0} \text{HINGE}(C\Phi - \text{BIAS}_{i,r})$ $X_{ij} \in \{0, \ldots, R\}$ $\text{BIAS}_{i,r}$ IS PER-ROW BIAS TERM	$R_k(C, \Phi) = \text{tr}(C\Phi')$ FAST MMMF: $\frac{1}{2}(\|C\|_F^2 + \|\Phi\|_F^2)$
METRIC MULTI-DIMENSIONAL SCALING (APP. C.2.4)	$\|K - C\Phi\|_F^2$ FOR ISOTROPIC KERNEL K	
NONNEGATIVE MATRIX FACTORIZATION	$\|X - C\Phi\|_F^2$	$C > 0$ $\Phi > 0$
NORMALIZED-CUT (PROP. 7)	$\left\|(\Lambda^{-1}X - C\Phi)\Lambda^{1/2}\right\|_F^2$	$\mathcal{C}_{\text{HARD}}$
PARTIAL LEAST SQUARES (APP. C.1.2)	$\|XY' - C\Phi\|_F^2$	
PCA (PROP. 2)	$\|X - C\Phi\|_F^2$	
EXP. FAMILY PCA (PROP. 4)	$D_{F^*}(C\Phi \| f^{-1}(X))$	
KERNEL PCA (PROP. 3)	$\|K - C\Phi\|_F^2$	
SPARSE PCA (SECTION 5.1)	$\|X - C\Phi\|_F^2$	$R_{\Phi,k}(\Phi) = \ell_0(\Phi)$
PROBABILISTIC LATENT SEMANTIC INDEXING (APP. C.1.1)	$\|X - C\Phi\|_F^2$	$C > 0$ $\Phi > 0$
RATIO CUT (APP. C.2.5)	$\|K - C\Phi\|_F^2$ FOR $K = L^\dagger$	

3.5 Summary

This chapter demonstrated the generality of regularized factor models, despite the simplicity of the formalism. Supervised learning and unsupervised learning were both cast as regularized factor models, with many different algorithms obtained depending on simple modeling choices, such as instance weighting, choice of kernel function and choice of loss function. This generality serves to motivate further algorithmic development for this class of problems, discussed in the next chapter. Moreover, more problems are unified under regularized factor models in Chapters 5 to 7 using different choices of regularizers. Finally, the connection between supervised and unsupervised learning established in this chapter enables the formulation of a principled semi-supervised learning algorithm in Section 6.1.

Chapter 4

Convex formulations for regularized factor models

In this chapter, I discuss how to reformulate a subclass of regularized factorization problems so that they are *globally* solvable (Zhang et al., 2011; White et al., 2012). The previous discussion focused on how to formalize problems as a regularized factor model whereas the next important question is how to actually solve the regularized factorization optimization. A natural solution approach is to use alternating descent approaches that alternate between the basis C and representation Φ. Though this can be effective, these non-convex alternations can lead to local minima and are often slow, which I discuss further in Section 4.1. If the problem can be formulated convexly, however, then we are guaranteed to converge to the global solution, regardless of initial settings. To date, the class of convexly reformulated matrix factorizations consists of norm regularizers on Φ and closed bounded sets on C, which is quite general; the set of tractably computable reformulations, however, remains quite small, as discussed in Section 4.3. As we will see in following chapters, this small but useful set enables convex reformulations and efficient solutions of some important problems.

There are two key components to the reformulation: 1) relaxing the hard rank constraint that *a priori* specifies k, the dimension of $\Phi \in \mathbb{R}^{k \times T}$, and 2) obtaining a regularizer directly on the low rank approximation, $Z = C\Phi$, induced by the chosen regularizers on Φ and C. The rank is instead determined by the optimization, where the variable $\Phi \in \mathbb{R}^{\infty \times T}$ has rank controlled by the optimization zeroing out many rows.

The main contribution of my work on convex matrix factorization is the second component of the reformulation: a characterization of the induced norm on $Z = C\Phi$ given the chosen regularizer on Φ. In particular, I provide an efficiently computable induced norm by restricting the class of regularizers on Φ to l_p-norms: $\{ \, || \cdot ||_{p,1} \mid 1 \leq p \leq \infty \}$. I use these convex solution approaches to tractably solve several important machine learning problems in later chapters, namely subspace

learning, semi-supervised learning and autoregressive moving average models.

Contributions:

Contribution 1. The most general set of choices, to date, for regularizers on the factors C and Φ that enables a convex reformulation of regularized factor models (Zhang et al., 2011). Previous reformulations were restricted to a row regularizer on the representation, Φ, and a column regularizer on the basis, C (Bach et al., 2008). I describe the more general set in Section 4.2.

Contribution 2. The most complete set of efficiently computable induced norms for the convex reformulation (Zhang et al., 2011; White et al., 2012). In particular, the first work (Zhang et al., 2011) extended beyond the previously known efficiently computable induced norms ℓ_1 and ℓ_2 to any ℓ_p norm on Φ and a partitioned constraint set on C. Moreover, the second work (White et al., 2012) further simplified the computation of the induced norm with a partitioned constraint set into an efficient line search over a weighted trace norm.

Contribution 3. A simple boosting algorithm for recovering solutions to the original problem from the convex reformulation, with a particularly efficient form for the ℓ_2 norm on Φ and partitioned constraints on C (White et al., 2012). The joint work (Zhang et al., 2011) also provides a closed form recovery for ℓ_p norms on Φ with an ℓ_1 norm on C and a closed form recovery for an ℓ_1 norm on Φ with a partitioned constraint set on C (see Proposition 8) for both of these novel closed form recoveries). Previous work was limited to a closed form recovery for ℓ_2 regularizers on both C and Φ and a closed form recovery for ℓ_1 on Φ (Bach et al., 2008).

4.1 Why not use expectation-maximization?

A natural question is why focus on convex reformulations when an expectation-maximization or alternating approach can be used. In fact, because global training procedures were generally not known for settings other than PCA, many regularized factorization algorithms were developed as alternating minimizations (Jenatton et al., 2010; Bradley and Bagnell, 2008; Elad and Aharon, 2006; Zou et al., 2006). The general expectation-maximization approach for matrix factorization is summarized in Algorithm 1, with both a hard Viterbi-EM step if Φ is part of the parameters $\boldsymbol{\theta}$ for the distribution $P(X|\boldsymbol{\theta})$ or a standard "expectation" step which computes the loss for $\boldsymbol{\theta}$ by marginalizing out Φ. The approach is intuitive and simple, and an argument could be made that the effort in finding convex reformulations is not worthwhile.

There are four main issues, however, with this non-global approach. First, finding local minima means we are not actually finding a solution to the specified objective. The global minima are the

Algorithm 1 Generic Expectation-Maximization for Factorized Models

Input: X, k and loss L

Initialize C and Φ randomly, or with some prior information
$L\left(C\Phi\ ;\ X\right) = -\log P(X|C,\Phi)$
while change in $L\left(C\ ;\ X\right) >$ tolerance **do**
  Hard E-Step: If Φ part of parameters θ:
    $\Phi = \operatorname{argmin}_{\Phi} L\left(C\Phi\ ;\ X\right)$
    $L\left(C\ ;\ X\right) = L\left(C\Phi\ ;\ X\right)$
  Marginal E-Step: If Φ not part of parameters $\boldsymbol{\theta}$ (i.e., unobserved latent data):
    $P(X_{:t}|C) = \int P(X_{:t}, \boldsymbol{\phi}_t|C)d\boldsymbol{\phi}_t = \int P(X_{:t}|\boldsymbol{\phi}_t, C)P(\boldsymbol{\phi}_t|C)d\boldsymbol{\phi}_t$
    $= \max_{q(\cdot)} \int q(\boldsymbol{\phi}_t)p(X_{:t}, \boldsymbol{\phi}_t|C)d\boldsymbol{\phi}_t + H(q(\cdot))$
    $L\left(C\ ;\ X\right) = -\sum_{t=1}^{T} \log P(X|C)$
  M-Step: $C = \operatorname{argmin}_{C} L\left(C\ ;\ X\right)$
end while

set of solutions to the objective. Local minima may not correspond to this set and so may not satisfy the properties specified by the objective. Second, algorithms that result in local minima can be very difficult to tune. Parameter settings can be changed, and the algorithm re-run; improvement in results, however, cannot be attributed to the parameter change, but rather could also be due to finding a different, better local minimum. This lack of controlled experimentation makes it difficult to select parameters, engineer useful features or understand the outcome from a given algorithm. Third, local minima also confound the validity of the objective. A specified objective may not properly enforce desired properties; if the objective is globally solvable, however, a vacuous or inappropriate objective can be discovered. For example, we demonstrate in Section 5.1 that, contrary to popular belief, the typical sparse coding optimization (Olshausen and Field, 1997) does not result in overcomplete codes. This insight results from the convex reformulation of the problem, that gives an exact solution for the sparse coding optimization. Local solvers, however, did not find the global, trivial vector-quantization solution, but rather spurious solutions that made it difficult to recognize the true outcome of the objective. Finally, alternating solutions are typically found to be slow (Redner and Walker, 1984). Computational gains have been reported numerous times from simple convex relaxations or reformulations of the problem (Cheng et al., 2013), and also in our own work shown in the empirical results over the next few chapters.

To find a convex reformulation, the only relaxation that is required is to relax the rank constraint. Instead of specifying k before the optimization, k is allowed to be arbitrarily large, with the chosen regularizers controlling the dimension. In this way, the optimization selects the appropriate rank for the problem. Though this can be described as relaxing the rank constraint, it can actually be seen as a useful way to avoid specifying the unknown rank *a priori*. In later chapters, we will see some of the effects of this "relaxation", particularly for subspace learning and sparse coding.

4.2 Convex matrix factorization formulation for norm regularizers

The approach to the following convex reformulations consists of two important steps:

1. A change of variables to $Z = C\Phi$, instead of optimizing for C and Φ directly.
2. A characterization of the *induced norm* $\|\!| \cdot |\!\|$: $\mathbb{R}^{n \times T} \rightarrow \mathbb{R}$ on Z that corresponds to the regularizers chosen on Φ and C.

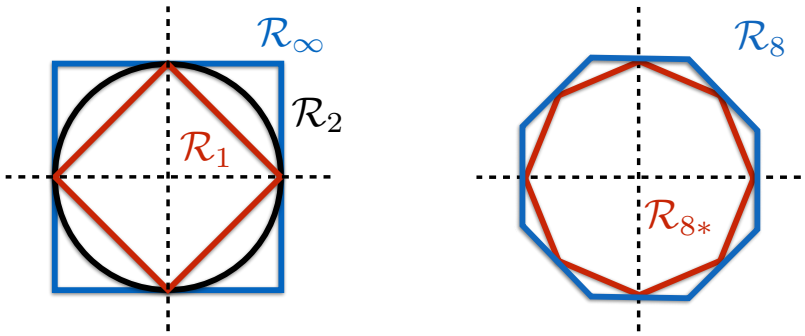
The difficulty is in the characterization of the induced regularizer on Z. For any regularizers on C and Φ, the corresponding regularizer on their product is not obvious, nor is it obvious that such an induced regularizer exists. To date, the regularization choices that result in known induced regularizers on Z are mostly restricted to *norm regularizers*. A norm $R : \mathcal{F} \rightarrow \mathbb{R}$ is a function that satisfies

1. $R(a\boldsymbol{\phi}) = |a|R(\boldsymbol{\phi})$
2. $R(\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2) \leq R(\boldsymbol{\phi}_1) + R(\boldsymbol{\phi}_2)$
3. $R(\boldsymbol{\phi}) = 0$ then $\boldsymbol{\phi} = \mathbf{0}$

Two key reasons that norm regularizers are so convenient are that 1) norms are convex and 2) for a general class (described in Theorem 2) of norms on Φ, a corresponding induced norm $\|\!|Z|\!\|$ for the product, $Z = C\Phi$, is both guaranteed to exist and to be convex. Since the induced norm is guaranteed to be convex, the resulting loss with regularizer $\|\!|Z|\!\|$ is also guaranteed to be convex. The induced norm, however, is not guaranteed to be efficiently computable. An important exploration, therefore, is to determine which induced norms can be computed efficiently, which we discuss further in Section 4.3.

In the joint work (Zhang et al., 2011; White et al., 2012), we generalized the class of regularizers on Φ to any $\{ \, || \cdot ||_{p,1} \mid 1 \leq p \leq \infty \}$ and any bounded closed set $\mathcal{C} \subset \mathbb{R}^n$ for $C_{:i} \in \mathcal{C}$ such that span$(\mathcal{C}) = \mathbb{R}^n$. This set includes any column norm regularizer on C, i.e., $C_{:i} \in \mathcal{C} = \{\mathbf{c} \in \mathbb{R}^n \mid \|\mathbf{c}\| \leq \beta\}$. The set of allowable regularizers on Φ has since been generalized to include any row norm regularizer, $\|\Phi\|_{\diamond,1}$; to the best of my knowledge, this is the most general class of convex reformulations of regularized factorization to date. Compared to (Bach et al., 2008), which allowed any row norm regularizer on Φ, this formulation is more general by allowing $C_{:i} \in \mathcal{C}$ for any bounded closed $\mathcal{C}$.

Lemma 3. *For any bounded closed set $\mathcal{C} \subset \mathbb{R}^n$ such that span$(\mathcal{C}) = \mathbb{R}^n$, and any vector norm $\|\mathbf{z}\|_\diamond$, the definition $\|X\|_{(\mathcal{C},\diamond)} = \max_{\mathbf{z} \in \mathcal{C}} \|X\mathbf{z}\|_\diamond$ establishes a norm on X.*



(a) ℓ_p norms (b) Norm duals for ℓ_p

Figure 4.1: Norms and their duals, where the norm value is shown for all $\mathbf{x} = (x_1, x_2)$ such that $\|\mathbf{x}\|_p = 1$. **(a)** ℓ_p norms. The dual of the ℓ_1 norm is ℓ_∞ and the dual of the ℓ_2 norm is itself. **(b)** The general relationship between the norm and its dual in ℓ_p spaces (here for $p = 8$).

Proof: It is easy to verify $\|X\|_{(\mathcal{C},\diamond)} \geq 0$ and $\|\alpha X\|_{(\mathcal{C},\diamond)} = |\alpha|\|X\|_{(\mathcal{C},\diamond)}$. Next, note

$$\|X+Y\|_{(\mathcal{C},\diamond)} = \max_{\mathbf{z}\in\mathcal{C}} \|(X+Y)\mathbf{z}\|_\diamond \leq \max_{\mathbf{z}\in\mathcal{C}} \|X\mathbf{z}\|_\diamond + \|Y\mathbf{z}\|_\diamond \leq \|X\|_{(\mathcal{C},\diamond)} + \|Y\|_{(\mathcal{C},\diamond)}.$$

Finally, if $\mathcal{C}$ is not restricted to a subspace of $\mathbb{R}^n$ then $\max_{\mathbf{z}\in\mathcal{C}} \|X\mathbf{z}\|_\diamond = 0$ implies $X = 0$. ■

Some useful examples of vector norms are ℓ_p norms, depicted in Figure 4.1. Norm duality is an important concept for finding an induced norm on the joint variable, Z, and for tractable computation described in the next section specifically for ℓ_p norms.

Theorem 2. *For any vector norm $\|\cdot\|_\diamond$ with conjugate $\|\cdot\|_{\diamond^*}$, and any bounded closed set $\mathcal{C} \subset \mathbb{R}^n$ such that* $\text{span}(\mathcal{C}) = \mathbb{R}^n$ *and $C \in \mathcal{C}^\infty$ indicating that C has an unrestricted number of columns,*

$$\min_{C\in\mathcal{C}^\infty} \min_{\Phi} L(C\Phi; X) + \alpha\|\Phi\|_{\diamond,1} \tag{4.1}$$

$$= \min_{Z} L(Z; X) + \alpha\|Z'\|^*_{(\mathcal{C},\diamond^*)} \tag{4.2}$$

with $\|Z'\|^*_{(\mathcal{C},\diamond^*)} = \max_{\|Y\|_{(\mathcal{C},\diamond^*)}\leq 1} \text{tr}(Y'Z')$ *for induced norm* $\|Y\|_{(\mathcal{C},\diamond^*)} = \max_{\mathbf{z}\in\mathcal{C}} \|Y\mathbf{z}\|_{\diamond^*}$ *defined in Lemma 3.*

Proof Sketch: [Full proof in Appendix D.2]

First, we do a change of variables to obtain

$$\begin{aligned}(4.1) &= \min_{Z} \min_{C\in\mathcal{C}^\infty} \min_{\Phi:C\Phi=Z} L(Z;X) + \alpha\|\Phi\|_{\diamond,1} \\ &= \min_{Z} L(Z;X) + \alpha \min_{C\in\mathcal{C}^\infty} \min_{\Phi:C\Phi=Z} \|\Phi\|_{\diamond,1}. \end{aligned} \tag{4.3}$$

Second, we show that the inner minimization in (4.3) defines an induced norm on Z, with key steps

1. Introduce Lagrange multiplier $\text{tr}(\Gamma'(Z - C\Phi))$ for the condition $\Phi : C\Phi = Z$, giving

$$\begin{aligned}\min_{\Phi:C\Phi=Z} \|\Phi\|_{\diamond,1} &= \min_{\Phi} \max_{\Gamma} \|\Phi\|_{\diamond,1} + \text{tr}(\Gamma'(Z - C\Phi)) \\ &= \max_{\Gamma} \min_{\Phi} \|\Phi\|_{\diamond,1} + \text{tr}(\Gamma'(Z - C\Phi)) \end{aligned} \tag{4.4}$$

where the min and max are swapped by showing that we satisfy Slater's condition, meaning strong duality holds (Boyd and Vandenberghe, 2004, §5.2.3).

2. Using norm duality, giving $\|\Phi\|_{\diamond,1} = \max_{V:\|V\|_{\diamond^*,\infty}\leq 1} \operatorname{tr}(V'\Phi)$, and algebraic simplifications, we obtain
$$(4.4) = \max_{\Gamma:\|C'\Gamma\|_{\diamond^*,\infty}\leq 1} \operatorname{tr}(\Gamma' Z).$$

3. By the definition of the norm in Lemma 3, we get that
$$\min_{C\in\mathcal{C}^\infty} \max_{\Gamma:\|C'\Gamma\|_{\diamond^*,\infty}\leq 1} \operatorname{tr}(\Gamma' Z) = \max_{\Gamma:\|\Gamma'\|_{(\mathcal{C},\diamond^*)}\leq 1} \operatorname{tr}(\Gamma' Z)$$

4. Finally, again by norm duality, $\max_{\Gamma:\|\Gamma'\|_{(\mathcal{C},\diamond^*)}\leq 1} \operatorname{tr}(\Gamma' Z) = \|Z'\|^*_{(\mathcal{C},\diamond^*)}$.

Therefore, $\min_{C\in\mathcal{C}^\infty} \min_{\Phi:C\Phi=Z} \|\Phi\|_{\diamond,1} = \|Z'\|^*_{(\mathcal{C},\diamond^*)}$, obtaining the desired result. ■

This theorem encompasses a surprising number of settings, due to the generality of the loss function and constraints on C and Φ, as we will see in its application to important problems over the next several chapters. The loss function can easily accommodate missing entries in X by restricting the loss evaluation to observed entries (Srebro et al., 2004). This theorem also applies to having a hard constraint on Φ and the norm regularizer on C, simply by taking transposes, as shown in the following corollary. Interestingly, therefore, at least one of the factors can be constrained at a given time, though potentially enforcing hard constraints on both simultaneously will lose the ability to obtain a convex reformulation.

Corollary 2. *For vector norm $\|\cdot\|_\diamond$, and any bounded closed set $\mathcal{F}\subset\mathbb{R}^T$ such that* $\operatorname{span}(\mathcal{F}) = \mathbb{R}^T$
$$\min_{\Phi\in\mathcal{F}^\infty\subseteq\mathbb{R}^{\infty\times T}} \min_C L(C\Phi;X) + \alpha\|C'\|_{\diamond,1} \;=\; \min_Z L(Z;X) + \alpha\|Z\|^*_{(\mathcal{C},\diamond^*)}. \tag{4.5}$$

Proof: Take the data matrix to be $X'\in\mathbb{R}^{T\times n}$ and $\mathcal{C}^\infty = \mathcal{F}^{\infty\prime}\subset\mathbb{R}^{T\times\infty}$. Then, we can apply our convex reformulation to obtain
$$\min_{C\in\mathcal{C}^\infty} \min_\Phi L(C\Phi;X') + \alpha\|\Phi\|_{\diamond,1} = \min_Z L(Z;X') + \alpha\|Z'\|^*_{(\mathcal{C},\diamond^*)}.$$
The solution to this convex optimization gives Z' for X and $Z = (Z')' = (C\Phi)' = \Phi' C'$. Let $\tilde{C} = \Phi'$, $\tilde{\Phi} = C'$, $\tilde{Z} = Z'$ and $\tilde{L}(Z,X) = L(Z',X')$. Then, since the loss remains convex in X

even on the transposed matrix,

$$\begin{aligned}\min_{\tilde{\Phi}\in\mathcal{F}^\infty\subseteq\mathbb{R}^{\infty\times T}} \min_{\tilde{C}} \tilde{L}(\tilde{C}\tilde{\Phi};X)+\alpha\|\tilde{C}'\|_{\diamond,1} &= \min_{\tilde{\Phi}'\in\mathcal{C}^\infty} \min_{\tilde{C}} \tilde{L}(\tilde{C}\tilde{\Phi};X)+\alpha\|\tilde{C}'\|_{\diamond,1} \\ &= \min_{C\in\mathcal{C}^\infty} \min_{\Phi} L(\Phi C;X')+\alpha\|\Phi\|_{\diamond,1} \\ &= \min_{Z} L(Z;X')+\alpha\|Z'\|^*_{(\mathcal{C},\diamond^*)} \\ &= \min_{Z} \tilde{L}(Z';X)+\alpha\|Z'\|^*_{(\mathcal{C},\diamond^*)} \\ &= \min_{\tilde{Z}} \tilde{L}(\tilde{Z};X)+\alpha\|\tilde{Z}\|^*_{(\mathcal{C},\diamond^*)}.\end{aligned}$$

■

4.3 Computationally practical special cases

Theorem 2 and Corollary 2 captures a wide range of formulations, including standard sparse coding and sparse PCA (Bradley and Bagnell, 2009; Bach et al., 2008; Zou et al., 2006), as we discuss in the next chapter. For (4.2) to admit an efficient global optimization procedure, however, the induced norm $\|Z'\|^*_{(\mathcal{C},\diamond^*)}$ must be efficiently computable for given Z. Unfortunately, although the induced norm is convex, it is not always efficiently computable (Hendrickx and Olshevsky, 2010; Steinberg, 2005). In particular, it is not known if this norm is efficiently computable for the mixed regularizers considered in (Bach et al., 2008; Bradley and Bagnell, 2009; Zou et al., 2006); hence these previous works had to introduce relaxations, heuristic basis generators, or alternating minimization, respectively.

Nevertheless, we will see that there remain many important and useful cases where $\|Z'\|^*_{(\mathcal{C},\diamond^*)}$ can be computed efficiently. Bach et al. (2008) showed the induced norm can be efficiently computed for l_1 and l_2 norms on Φ and C. We generalize these results to include efficient closed form induced norms for l_p regularizers and for a block constraint C, where the top and bottom partition of C can be constrained separately. The following theorems indicate five cases where the induced regularizer has a closed form, making it amenable to practical application.

Theorem 3. *For regularizer $\|\Phi\|_{2,1}$ and constraint set $\mathcal{C}_2 = \{\mathbf{c} : \|\mathbf{c}\|_2 \leq 1\}$, the induced norm on Z is*

$$\|Z'\|^*_{(\mathcal{C}_2,2)} = \|Z\|_{tr}.$$

Proof: $\|Z'\|_{(\mathcal{C}_2,2)} = \max_{\|\mathbf{c}\|_2\leq 1}\|Z'\mathbf{c}\|_2 = \|Z'\|_{sp}$ by definition. Since the dual of the spectral norm is the trace norm, we have $\|Z'\|^*_{(\mathcal{C}_2,2)} = \|Z'\|^*_{sp} = \|Z'\|_{tr}$. ■

Theorem 4. *For regularizer $\|\Phi\|_{p,1}$ and constraint set $\mathcal{C}_1 = \{\mathbf{c} : \|\mathbf{c}\|_1 \leq 1\}$ the induced norm on Z is*

$$\|Z'\|^*_{(\mathcal{C}_1,p^*)} = \|Z\|_{p,1}.$$

Proof: We simply need to show that $\|Y\|^*_{(1,r)} = \|Y'\|_{r^*,1}$ for any $1 \le r \le \infty$, $\frac{1}{r} + \frac{1}{r^*} = 1$.

$$\begin{aligned}
\|Y\|^*_{(1,r)} &= \max_{C:\|C\|_{(1,r)}\le 1} \operatorname{tr}(C'Y) && \triangleright \text{ by definition} \\
&= \max_{C:\|C_{:t}\|_r\le 1} \sum_t C'_{:t}Y_{:t} && \triangleright \text{ because } \|C\|_{(1,r)} = \max_t \|C_{:t}\|_r \text{ (Steinberg, 2005, §1.3.1)} \\
&= \sum_t \|Y_{:t}\|_{r^*} && \triangleright \text{ by Hölder's inequality, for the extremal equality case} \\
&= \|Y'\|_{r^*,1}.
\end{aligned}$$

Therefore, $\|Z'\|^*_{(\mathcal{C}_1,p^*)} = \|Z'\|^*_{(1,p^*)} = \|Z\|_{p,1}$. ■

Theorem 5. *For regularizer $\|\Phi\|_{1,1}$ and constraint set $\mathcal{C}_q = \{\mathbf{c} : \|\mathbf{c}\|_q \le 1\}$ the induced norm on Z is*

$$\|Z'\|^*_{(\mathcal{C}_q,\infty)} = \|Z'\|_{q,1}.$$

Proof:

$$\begin{aligned}
\|Z'\|^*_{(\mathcal{C}_q,\infty)} = \|Z'\|^*_{(q,\infty)} = \|Z\|^*_{(1,q^*)} && \triangleright \text{ because } \|Y'\|_{(q^*,p^*)} = \|Y\|_{(p,q)} \text{ (Steinberg, 2005, §1.2.2)} \\
= \|Z'\|_{q,1} && \triangleright \text{ because } \|Y\|^*_{(1,r)} = \|Y'\|_{r^*,1} \text{ shown in Theorem 4.}
\end{aligned}$$ ■

Theorem 6. *For regularizer $\|\Phi\|_{1,1}$ and $\mathcal{C}_{q_1q_2} = \left\{\mathbf{c} = \begin{bmatrix}\mathbf{c}^{(1)}\\ \mathbf{c}^{(2)}\end{bmatrix} : \|\mathbf{c}^{(1)}\|_{q_1} \le \beta_1, \|\mathbf{c}^{(2)}\|_{q_2} \le \beta_2\right\}$ the induced norm on Z is*

$$\|Z'\|^*_{(\mathcal{C}_{q_1q_2},\infty)} = \sum_t \max\left(\frac{1}{\beta_1}\|Z^{(1)}{}_{:t}\|_{q_1}, \frac{1}{\beta_2}\|Z^{(2)}{}_{:t}\|_{q_2}\right).$$

Proof: Let Γ be the dual variable, which is of the same size as Z. First, we determine the dual norm on Z, which simplifies

$$\begin{aligned}
\|\Gamma'\|_{(\mathcal{C}_{q_1q_2},\infty)} &= \max_{\mathbf{c}\in\mathcal{C}_{q_1q_2}} \|\Gamma'\mathbf{c}\|_\infty = \max_{\mathbf{c}\in\mathcal{C}_{q_1q_2}} \max_t \Gamma'_{:t}\mathbf{c} \\
&= \max_t \max_{\mathbf{c}^{(1)}:\|\mathbf{c}^{(1)}\|_{q_1}\le\beta_1} \max_{\mathbf{c}^{(2)}:\|\mathbf{c}^{(2)}\|_{q_2}\le\beta_2} \left[\mathbf{c}^{(1)\prime}\Gamma^{(1)}{}_{:t} + \mathbf{c}^{(2)\prime}\Gamma^{(2)}{}_{:t}\right] \\
&= \max_t \left[\max_{\mathbf{c}^{(1)}:\|\mathbf{c}^{(1)}\|_{q_1}\le\beta_1} \mathbf{c}^{(1)\prime}\Gamma^{(1)}{}_{:t} + \max_{\mathbf{c}^{(2)}:\|\mathbf{c}^{(2)}\|_{q_2}\le\beta_2} \mathbf{c}^{(2)\prime}\Gamma^{(2)}{}_{:t}\right] \\
&= \max_t \left[\max_{\mathbf{c}^{(1)}:\|\mathbf{c}^{(1)}\|_{q_1}\le 1} \beta_1\mathbf{c}^{(1)\prime}\Gamma^{(1)}{}_{:t} + \max_{\mathbf{c}^{(2)}:\|\mathbf{c}^{(2)}\|_{q_2}\le 1} \beta_2\mathbf{c}^{(2)\prime}\Gamma^{(2)}{}_{:t}\right] && \triangleright \text{ change of variables} \\
&= \max_t \left[\beta_1\|\Gamma^{(1)}{}_{:t}\|_{q_1^*} + \beta_2\|\Gamma^{(2)}{}_{:t}\|_{q_2^*}\right] && \triangleright \text{ by norm duality.}
\end{aligned}$$

With this form for the dual norm, we can compute the norm on Z more easily, because the dual of the max-norm is the 1-norm. Therefore, the inner sum becomes a max and the outer max becomes a sum. To simplify notation, let $A_{t,i} = \beta_i\|\Gamma^{(i)}_{:,t}\|_{q_i^*}$, giving $A \in \mathbb{R}^{T\times 2}$. Then

$$\|\Gamma'\|_{(\mathcal{C}_{q_1q_2},\infty)} = \max_t \beta_1\|\Gamma^{(1)}{}_{:t}\|_{q_1^*} + \beta_2\|\Gamma^{(2)}{}_{:t}\|_{q_2^*} = \max_t[A_{t,1} + A_{t,2}] = \|A\|_{\infty,1}$$

and

$$
\begin{aligned}
\|Z'\|^*_{(\mathcal{C}_{q_1 q_2},\infty)} &= \max_{\Gamma:(\max_t \beta_1\|\Gamma^{(1)}{}_{:t}\|_{q_1^*}+\beta_2\|\Gamma^{(2)}{}_{:t}\|_{q_2^*})\le 1} \operatorname{tr}(\Gamma' Z) \\
&= \max_{A:\|A\|_{\infty,1}\le 1} \max_{\Gamma:\beta_i\|\Gamma^{(i)}_{:,t}\|_{q_i^*}\le A_{t,i}} \sum_{t,i} \Gamma^{(i)'}{}_{:,t} Z_{:,t} \\
&= \max_{A:\|A\|_{\infty,1}\le 1} \sum_{t,i} \max_{\Gamma:\|\Gamma^{(i)}_{:,t}\|_{q_i^*}\le 1} \frac{A_{t,i}}{\beta_i} \Gamma^{(i)'}{}_{:,t} Z_{:,t} \\
&= \max_{A:\|A\|_{\infty,1}\le 1} \sum_{t,i} \frac{A_{t,i}}{\beta_i} \max_{\|\Gamma^{(i)}_{:,t}\|_{q_i^*}\le 1} \Gamma^{(i)'}{}_{:,t} Z_{:,t} && \triangleright \text{ by norm duality} \\
&= \max_{A:\|A\|_{\infty,1}\le 1} \sum_{t,i} \frac{A_{t,i}}{\beta_i} \|Z_{:,t}\|_{q_i} \\
&= \max_{A:\|A\|_{\infty,1}\le 1} \sum_{t,i} A_{t,i} B_{t,i} && \triangleright \text{ for } B_{t,i} = \frac{1}{\beta_i}\|Z^{(i)}_{:,t}\|_{q_i} \\
&= \max_{A:\|A\|_{\infty,1}\le 1} \sum_{t,i} \operatorname{tr}(A'B) = \|B\|_{1,\infty} && \triangleright \text{ by norm duality} \\
&= \sum_t \max\left(\frac{1}{\beta_1}\|Z^{(1)}{}_{:t}\|_{q_1}, \frac{1}{\beta_2}\|Z^{(2)}{}_{:t}\|_{q_2}\right)
\end{aligned}
$$

■

Theorem 7. *For regularizer* $\|\Phi\|_{2,1}$ *and* $\mathcal{C}_{2,2} = \{\mathbf{c} = \begin{bmatrix}\mathbf{c}^{(1)}\\ \mathbf{c}^{(2)}\end{bmatrix} : \|\mathbf{c}^{(1)}\|_2 \le \beta_1, \|\mathbf{c}^{(2)}\|_2 \le \beta_2\}$ *the induced norm on* Z *is*

$$\|Z'\|^*_{(\mathcal{C}_{2,2},2)} = \max_{0\le\eta\le 1} \|E_\eta^{-1} Z\|_{tr},$$

where

$$E_\eta := \begin{bmatrix} \beta_1/\sqrt{\eta}\, I_n & 0 \\ 0 & \beta_2/\sqrt{1-\eta}\, I_n \end{bmatrix}.$$

Moreover $\left\|E_\eta^{-1} Z\right\|_{tr}$ *is concave in* η *over* $[0,1]$*, enabling an efficient line-search for the norm.*

Proof Sketch: [Full proof in Appendix D.3] The proof consists of

1. characterizing the norm $\|\Gamma\|_{(\mathcal{C}_{2,2},2)} = \min_{\rho\ge 0} \|E_{\beta_1^2/(\beta_2^2\rho+\beta_1^2)}\Gamma\|_{\text{sp}}$
2. characterizing the conjugate norm of $\|\Gamma\|_{(\mathcal{C}_{2,2},2)}$

$$\|Z\|^*_{(\mathcal{C}_{2,2},2)} = \max_{\Gamma:\|\Gamma\|_{(\mathcal{C}_{2,2},2)}\le 1} \operatorname{tr}(\Gamma' Z) = \max_{\rho\ge 0} \|E^{-1}_{\beta_1^2/(\beta_2^2\rho+\beta_1^2)} Z\|_{\text{sp}}$$

3. finally re-parametrizing with $\rho = \frac{\beta_1^2(1-\eta)}{\beta_2^2\eta}$ to get $E_{\beta_1^2/(\beta_2^2\rho+\beta_1^2)} = E_\eta$ for $0 \le \eta \le 1$.

■

As we will see in later chapters, the $(2,1)$-block norm is generally useful because it controls the rank of k; otherwise, the factorization might not be particularly meaningful. This norm controls the rank by pushing as many rows to zero as possible, while only smoothing the columns (Argyriou et al., 2008). This property is further justified in the next chapter in Section 5.2.

These proofs extend to a few other cases due to three equivalent formulations for the induced norm (Bach et al., 2008). From Equations 2,3 and 4 in (Bach et al., 2008), we know that the induced norm on Z can be written in several ways given the norm regularizers on the columns of C, $\|C_{:i}\|_C$, and the rows of Φ, $\|\Phi_{i:}\|_R$. We let $\|\!|Z|\!\| = \|Z'\|^*_{C,\diamond^*}$ represent the induced norm on Z when the context is clear. Then

$$\|\!|Z|\!\| = \min_{C,\Phi:Z=C\Phi} \frac{1}{2}\sum_{i=1}^{k}(\|\Phi_{i:}\|_R^2 + \|C_{:i}\|_C^2) \tag{4.6}$$

$$= \min_{C,\Phi:Z=C\Phi} \sum_{i=1}^{k} \|\Phi_{i:}\|_R\|C_{:i}\|_C \tag{4.7}$$

$$= \min_{C,\Phi:Z=C\Phi,\ \forall i\ \|C_{:i}\|_C=1} \sum_{i=1}^{k} \|\Phi_{i:}\|_R \tag{4.8}$$

$$= \min_{C,\Phi:Z=C\Phi,\ \forall i\ \|\Phi_{i:}\|_R=1} \sum_{i=1}^{k} \|C_{:i}\|_C. \tag{4.9}$$

The step between (4.7) and (4.8) is clear since C and Φ can be rescaled as $C_{:i}/\|C_{:i}\|_C$ and $\Phi_{i:}\|C_{:i}\|_C$, making $\|C_{:i}\|_C = 1$ in (4.7). Similarly, by rescaling with $\|\Phi_{i:}\|_R$, we can get the equivalence between (4.7) and (4.9). The tricky step is the second one, showing (4.6) = (4.7). Take $s_i = \frac{\|\Phi_{i:}\|_R}{\|C_{:i}\|_C}$. Then, since $Z = C\Phi = C\,\mathrm{diag}(\mathbf{s})\,\mathrm{diag}(\mathbf{s}^{-1})\Phi$, we can take the minimum of (4.6) and rescale to obtain

$$\|\!|Z|\!\| = \|\!|C\,\mathrm{diag}(\mathbf{s})\,\mathrm{diag}(\mathbf{s}^{-1})\Phi|\!\| = \frac{1}{2}\sum_{i=1}^{k}(\|\Phi_{i:}/s_i\|_R^2 + \|s_iC_{:i}\|_C^2) \text{ for all } \mathbf{s} = [s_1, \ldots, s_k].$$

Therefore, the original minimization could have any $\mathbf{s}$ scaling

$$\|\!|Z|\!\| = \min_{C,\Phi:Z=C\Phi} \frac{1}{2}\sum_{i=1}^{k}(\|\Phi_{i:}/s_i\|_R^2 + \|s_iC_{:i}\|_C^2)$$

including for $s_i = \frac{\|\Phi_{i:}\|_R}{\|C_{:i}\|_C}$, giving

$$\begin{aligned}
\|\!|Z|\!\| &= \min_{C,\Phi:Z=C\Phi} \frac{1}{2}\sum_{i=1}^{k}(\|\Phi_{i:}/s_i\|_R^2 + \|s_iC_{:i}\|_C^2) \\
&= \min_{C,\Phi:Z=C\Phi} \frac{1}{2}\sum_{i=1}^{k}(\|\Phi_{i:}\|_R\|C_{:i}\|_C + \|C_{:i}\|_C\|\Phi_{i:}\|_R) \\
&= \min_{C,\Phi:Z=C\Phi} \sum_{i=1}^{k} \|\Phi_{i:}\|_R\|C_{:i}\|_C.
\end{aligned}$$

Notice, however, that any recovered Φ and C may not be equivalent between the three formulations, as seen in Section 7.4. (4.9) was not originally in (Bach et al., 2008), but clearly follows from (4.7).

As before, this induced norm $\|\!|Z|\!\|$ is only convex when $k \to \infty$, where the regularizer implicitly controls the size of k. These equivalences give the following corollary.

Corollary 3. *The following result from Theorems 3-7, respectively.*

$$\|\Phi_{i:}\|_R = \|\Phi_{i:}\|_2,\ \|C_{:i}\|_C = \|C_{:i}\|_2 \ \mapsto\ \ \|\!|Z|\!\| = \|Z\|_{tr} \tag{4.10}$$

$$\|\Phi_{i:}\|_R = \|\Phi_{i:}\|_1,\ \|C_{:i}\|_C = \|C_{:i}\|_q \ \mapsto\ \ \|\!|Z|\!\| = \|Z'\|_{q,1} \tag{4.11}$$

$$\|\Phi_{i:}\|_R = \|\Phi_{i:}\|_p,\ \|C_{:i}\|_C = \|C_{:i}\|_1 \ \mapsto\ \ \|\!|Z|\!\| = \|Z\|_{p,1} \tag{4.12}$$

$$\|\Phi_{i:}\|_R = \|\Phi_{i:}\|_1,\ \|C_{:i}\|_C = \max(\|{C^{(1)}}_{:i}\|_{q_1}, \|{C^{(2)}}_{:i}\|_{q_2}) \\ \mapsto\ \ \|\!|Z|\!\| = \sum_t \max\left(\frac{1}{\beta_1}\|{Z^{(1)}}_{:t}\|_{q_1}, \frac{1}{\beta_2}\|{Z^{(2)}}_{:t}\|_{q_2}\right) \tag{4.13}$$

$$\|\Phi_{i:}\|_R = \|\Phi_{i:}\|_2,\ \|C_{:i}\|_C = \max(\|{C^{(1)}}_{:i}\|_2, \|{C^{(2)}}_{:i}\|_2) \\ \mapsto\ \ \|\!|Z|\!\| = \max_{0\le\eta\le 1} \left\|E_\eta^{-1} Z\right\|_{tr}. \tag{4.14}$$

4.3.1 Recovery algorithms

Once the global Z has been computed, the original factors C and Φ may be desired as well. In certain settings, such as transductive learning, Z may be all that is required; for many cases, however, the original goal was to explicitly find factors C and Φ, rather than just their product.

For certain regularization choices, there is a closed form recovery of C and Φ, which we provide in the next proposition. For other situations, we have developed a general boosting algorithm that enables the columns and rows of C and Φ, respectively, to be generated iteratively and reweighted until $C\Phi = Z$. The boosting procedure requires an efficient approach to generate these columns and rows; fortunately, for the above closed form formulations, these "oracle" directions that guarantee improvement in the solution are known. We present the general algorithm in Algorithm 2, and the oracle in Algorithm 3, which is the same for both the Frobenius norm case and for the $(2,1)$-block norm. The resulting rescaling, however, is different and is given in Algorithms 5 and 6.

Proposition 8. *Given optimal Z, there is a closed form recovery of the optimal C and Φ for the following four cases.*

1. *For $\|\Phi\|_{2,1}$, $\mathcal{C}_2 = \{\mathbf{c} : \|\mathbf{c}\|_2 \le 1\}$, giving $\|\!|Z|\!\| = \|Z\|_{tr}$, the recovery requires a simple singular value decomposition: $Z = U\Sigma V'$ gives $C = U$ and $\Phi = \Sigma V'$.*

2. *For $\|\Phi\|_F^2$, $\|C\|_F^2$, giving $\|\!|Z|\!\| = \|Z\|_{tr}$, the recovery requires a simple singular value decomposition: $Z = U\Sigma V'$ gives $C = U\sqrt{\Sigma}$ and $\Phi = \sqrt{\Sigma}V'$.*

3. *For $\|\Phi\|_{1,1}, \mathcal{C}_q = \{\mathbf{c} : \|\mathbf{c}\|_q \le 1\}$, giving $\|\!|Z|\!\| = \|Z'\|_{q,1}$, the recovery is $C = [Z_{:,1}/\|Z_{:,1}\|_q, \ldots, Z_{:,T}/\|Z_{:,T}\|_q]$ and $\Phi = \mathrm{diag}(\|Z_{:,1}\|_q, \ldots, \|Z_{:,T}\|_q)$.*

Algorithm 2 Generic Boosting Recovery Algorithm)

Input: Z, induced norm $\|\!| \cdot |\!\|$, Oracle algorithm and Rescaling algorithm
Output: C, Φ such that $C\Phi = Z$

1: $l(M)$: return $\|Z - M\|_F^2$ // define boosting loss
2: $C = [\,],\ \Phi = [\,]$ // Initialize C and Φ to empty matrices
3: $s = \|\!| Z |\!\|$
4: $Z = Z/s$ // The solution is properly rescaled after the procedure
5: $M_0 = \mathbf{0}$ // matrix of all zeros the same size as Z
6: **while** $l(M_{k-1}) <$ tolerance, for $k = 1, 2, \ldots$ **do**
7: // **Weak learning "oracle" step:** Greedily pick $(\mathbf{c}_k, \boldsymbol{\phi}_k)$
8: $(\mathbf{c}_k, \boldsymbol{\phi}_k)$ = OracleAlgorithm(Z, M_{k-1})
9: // **"Totally corrective" step:** Reweight the generated bases
10: $\boldsymbol{\mu}^{(k)} = \underset{\boldsymbol{\mu} \geq \mathbf{0}, \sum_i \mu_i = 1}{\operatorname{argmin}} \; l\Big(\sum_{i=1}^{k} \mu_i \mathbf{c}_i \boldsymbol{\phi}_i'\Big)$
11: $M_k = \sum_{i=1}^{k} \mu_i^{(k)} \mathbf{c}_i \boldsymbol{\phi}_i'$.
12: **end while**
13: // Set scales to preserve constraint/regularizer equivalence to $\|\!| Z |\!\|$
14: $(\mathbf{s}_1, \mathbf{s}_2)$ = RescalingAlgorithm($sZ, \boldsymbol{\mu}$)
15: $C = [\mathbf{c}_1, \ldots, \mathbf{c}_k] \operatorname{diag}(\mathbf{s}_1)$
16: $\Phi = \operatorname{diag}(\mathbf{s}_2)[\boldsymbol{\phi}_1; \ldots; \boldsymbol{\phi}_k]$.
17: **return** (C, Φ)

4. *For* $\|\Phi\|_{1,1}$, $\mathcal{C}_{q_1,q_2} = \{\mathbf{c} : \|\mathbf{c}^{(1)}\|_{q_1} \leq \beta_1, \|\mathbf{c}^{(2)}\|_{q_2} \leq \beta_2\}$, *giving* $\|\!| Z |\!\| = \sum_t \max\left(\frac{1}{\beta_1}\|Z^{(1)}{}_{:t}\|_{q_1}, \frac{1}{\beta_2}\|Z^{(2)}{}_{:t}\|_{q_2}\right)$, *the recovery is* $\Phi_{t,t} = \max\left(\frac{1}{\beta_1}\|Z^{(1)}{}_{:t}\|_{q_1}, \frac{1}{\beta_2}\|Z^{(2)}{}_{:t}\|_{q_2}\right)$ *and* $C = Z\Phi^{-1}$.

Proof: For 1, clearly $C\Phi = Z$ and, since V is orthogonal, making $(V_{:,i})'V_{:,i} = 1$

$$\|\Phi\|_{2,1} = \sum_{i=1}^{k} \|\Sigma_{ii} V_{:,i}\|_2 = \sum_{i=1}^{k} \Sigma_{ii} \|V_{:,i}\|_2 = \sum_{i=1}^{k} \Sigma_{ii} \sqrt{(V_{:,i})'V_{:,i}} = \sum_{i=1}^{k} \Sigma_{ii} = \|Z\|_{tr}$$

For 2, again $C\Phi = Z$ and, since the Froebnius norm is invariant under transformation by an orthogonal matrix V

$$\frac{1}{2}(\|\Phi\|_F^2 + \|C\|_F^2) = \frac{1}{2}(\left\|\sqrt{\Sigma}\right\|_F^2 + \left\|\sqrt{\Sigma}\right\|_F^2) = \operatorname{tr}(\Sigma) = \|Z\|_{tr}$$

For 3, again $C\Phi = Z$ since the normalization is removed in the product, and

$$\|\Phi\|_{1,1} = \|\operatorname{diag}(\|Z_{:,1}\|_q, \ldots, \|Z_{:,T}\|_q)\|_{1,1} = \sum_{t=1}^{T} \|Z_{:,t}\|_q = \|Z'\|_{q,1}$$

For 4, the procedure is the same as for 3. ■

For other convex reformulations, there is no closed form recovery. For this situation, we provide a boosting procedure which consists of the following four steps, summarized in Algorithm 2:

Recovery boosting algorithm given column norm $\|\cdot\|_C$ and row norm $\|\cdot\|_R$:

1. Normalize Z such that $\|\!|Z|\!\| = 1$; after recovering C and Φ, we rescale using the original $\|\!|Z|\!\|$.

2. For any matrices C and Φ, we can write $C = [\mathbf{c}_1, \ldots, \mathbf{c}_k]\,\mathrm{diag}(\mathbf{s}_1)$ and $\Phi = \mathrm{diag}(\mathbf{s}_2)[\boldsymbol{\phi}_1; \ldots; \boldsymbol{\phi}_k]$ for $\|\mathbf{c}_i\|_C = 1$ and $\|\boldsymbol{\phi}_i\|_R = 1$ and scale vectors $\mathbf{s}_1, \mathbf{s}_2 \geq 0$. This representation simplifies generation of C and Φ.

3. Generate unit vectors and scales to optimize $l(M) = \|Z - M\|_F^2$, i.e.,

$$\min_{C,\Phi} \|Z - C\Phi\|_F^2 = \min_{\mathbf{c}_1,\mathbf{c}_2,\ldots,\mathbf{s}_1,\mathbf{s}_2,\boldsymbol{\phi}1,\boldsymbol{\phi}2,\ldots,\mathbf{s}_2} \|Z - C\Phi\|_F^2$$

in a repeated two step boosting approach, starting with $M_0 = 0$:

3.1 Weak learning "oracle" step: greedily pick

$$(\mathbf{c}_k, \boldsymbol{\phi}_k) \in \operatorname*{argmin}_{\|\mathbf{c}\|_C=1, \|\boldsymbol{\phi}\|_R=1} \left\langle \nabla l(M_{k-1}), \mathbf{c}_k \boldsymbol{\phi}_k' \right\rangle$$

In general, the oracle will be different for different norms $\|\cdot\|_C$ and $\|\cdot\|_R$. If they are both the ℓ_2 norm, for example, then the oracle step can be computed efficiently by using the top singular vectors of the residual $R = Z - M_{k-1} = \nabla l(M_{k-1})$

$$\operatorname*{argmax}_{\|\mathbf{c}\|_2=1, \|\boldsymbol{\phi}\|_2=1} \mathbf{c}' R \boldsymbol{\phi} \implies \mathbf{c} = U_{:,1}, \boldsymbol{\phi} = V_{:,1} \text{ for } R = U\Sigma V'.$$

To see why, we can write the Lagrange equation which ensures that $\mathbf{c}$ and $\boldsymbol{\phi}$ are unit vectors: $\mathbf{c}' R\boldsymbol{\phi} - \lambda_1 \mathbf{c}'\mathbf{c} - \lambda_2 \boldsymbol{\phi}'\boldsymbol{\phi}$, for Lagrange multipliers λ_1 and λ_2. Taking the gradient with respect to $\mathbf{c}$ and $\boldsymbol{\phi}$ and setting the gradients to zero to obtain the optimal unit vectors, we get the two equations:

$$R\boldsymbol{\phi} = 2\lambda_1 \mathbf{c} \implies \mathbf{c}' R\boldsymbol{\phi} = 2\lambda_1 \mathbf{c}'\mathbf{c} = 2\lambda_1$$
$$R'\mathbf{c} = 2\lambda_2 \boldsymbol{\phi} \implies \boldsymbol{\phi}' R'\mathbf{c} = 2\lambda_2 \boldsymbol{\phi}'\boldsymbol{\phi} = 2\lambda_2$$

This implies that $\sigma_1 = 2\lambda_1 = 2\lambda_2$, giving $R\boldsymbol{\phi} = \sigma_1 \mathbf{c}$ and $R'\mathbf{c} = \sigma_1 \boldsymbol{\phi}$, which by definition are the left and right singular vectors of R. For ℓ_2 norms, however, we have a closed form recovery; this approach is only useful if a complete SVD needs to be avoided. In Algorithm 3, we provide an oracle for the partitioned case which does not have a closed form recovery.

3.2 "Totally corrective" step:

$$\boldsymbol{\mu}^{(k)} = \operatorname*{argmin}_{\boldsymbol{\mu} \geq \mathbf{0}, \sum_i \mu_i = 1} l\Big(\sum_{i=1}^{k} \mu_i \mathbf{c}_i \boldsymbol{\phi}_i'\Big) \quad \to \quad M_k = \sum_{i=1}^{k} \mu_i^{(i)} \mathbf{c}_i \boldsymbol{\phi}_i'.$$

Notice that the scales are reoptimized after each basis is added, meaning that bases that are not as useful will have their scale set to zero. This procedure will find a M_k satisfying $\|Z - M_k\|_F^2 < \epsilon$ within $O(1/\epsilon)$ iterations (Shalev-Shwartz et al., 2010, Theorem 2.7), since the ℓ_2-norm is 1-smooth.

4. Set $C = [\mathbf{c}_1, \ldots, \mathbf{c}_k] \operatorname{diag}(\mathbf{s}_1)$ and $\Phi = \operatorname{diag}(\mathbf{s}_2)[\boldsymbol{\phi}_1; \ldots; \boldsymbol{\phi}_k]$, where the scales $\mathbf{s}_1$ and $\mathbf{s}_2$ are determined according to the chosen constraints and regularizers to ensure the joint regularizer $\|\!|Z|\!\|$ equals the corresponding regularizer(s) for C and Φ. For example, if the optimization constrains C in the set $\|\mathbf{c}\|_C \leq 1$, then the scale should all be placed on Φ: $\mathbf{s}_1 = \mathbf{1}$ and $\mathbf{s}_2 = \|\!|Z|\!\|\boldsymbol{\mu}$, giving

$$\|\Phi\|_R = \|\operatorname{diag}(\mathbf{s}_2)[\boldsymbol{\phi}_1; \ldots; \boldsymbol{\phi}_k]\|_R = \|\!|Z|\!\| \sum_{i=1}^{k} \mu_i \|[\boldsymbol{\phi}_1; \ldots; \boldsymbol{\phi}_k]\|_R = \|\!|Z|\!\| \sum_{i=1}^{k} \mu_i = \|\!|Z|\!\|$$

since $\sum_{i=1}^{k} \mu_i = 1$ by the constraints in step 3.2. See Algorithms 5 and 6 for two rescalings.

Algorithm 3 Oracle for $\|\!|Z|\!\| = \max_{0 \leq \eta \leq 1} \|E_\eta^{-1} Z\|_{tr}$

Input: Z, M_{k-1}
Output: $\mathbf{c}, \boldsymbol{\phi}$, as new column and row to add to C, Φ respectively

// Note that this oracle uses a different loss function, l, which should be used in place of the
// l in Algorithm 2; see Appendix D.4 for derivation of the following oracle
$[U, \sim, V] = \operatorname{svd}(E_\eta^{-1} Z)$
$\Gamma = E_\eta^{-1} U V'$
$l(M)$: return $\|M\Gamma - Z\|_F^2$
// Compute the matrix where $\langle R, \mathbf{c}\mathbf{c}' \rangle = 0$
$\rho = \frac{(1-\eta)\beta_1}{\eta \beta_2^2}$
$\lambda_1 = 1/(\beta_1^2 + \beta_2^2 \rho)$
$\lambda_2 = \lambda_1 \rho$
$R = \lambda_1 I + \lambda_2 I - \Gamma\Gamma'$
$N =$ Algorithm 4 (R) // Orthonormal basis of nullspace(R)
$N_1 = N(1 : n_1, :),\ N_2 = N((n_1 + 1) : (n_1 + n_2), :)$ // the two partitions of N
$[V, \Sigma] = \operatorname{eigs}(\beta_2^2 N_1' N_1 - \beta_1^2 N_2' N_2)$
$Q = 2(Z - M_{k-1}\Gamma)\Gamma'$
$T = V' N' Q N V$
// Line search over τ over convex function $\lambda_{\max}$ which gives maximum eigenvalue
[f,g] = λloss(τ):
    $[\lambda, \mathbf{v}] = \lambda_{\max}(T - \tau\Sigma)$
    $\mathrm{f} = \lambda$
    $\mathrm{g} = -\Sigma \mathbf{v}^2$
[τ, f] = lbfgs(λloss, $\tau_{\text{init}} = 0$) // Apply your favourite non-smooth optimizer
// Now recover $\mathbf{c}$
G = Algorithm 4 (f$I + \tau\Sigma - T$) // Orthonormal basis of nullspace(f$I + \tau\Sigma - T$)
$[\lambda, \mathbf{v}] = \operatorname{eigs}(\mathrm{G}'\Sigma\mathrm{G})$
$\mathbf{c} = \mathrm{G}\,\mathbf{v}$
return $(\mathbf{c}, \mathbf{c})$

4.4 Summary

This chapter presented the main algorithmic advances for solving regularized factor models. The approach consists of finding a convex reformulation of biconvex objectives, which is not jointly

Algorithm 4 Orthonormal basis of nullspace

Input: R

Output: N orthonormal basis of the nullspace of R

1: $[\sim, \Sigma, V] = \text{svd}(R)$
2: $r = \text{count}(\Sigma > 1e-4)$ // rank of R
3: $N = V(:, (r+1) : n-r)$ // Last $n-r$ columns where V has n columns
4: **return** N

Algorithm 5 Rescaling for $\|\Phi\|_R = \|\Phi\|_{2,1}$, $\mathcal{C} = \{\mathbf{c} = \begin{bmatrix}\mathbf{c}^{(1)} \\ \mathbf{c}^{(2)}\end{bmatrix} : \|\mathbf{c}^{(1)}\|_2 \leq \beta_1, \|\mathbf{c}^{(2)}\|_2 \leq \beta_2\}$

Input: $Z, \boldsymbol{\mu}$

Output: $\mathbf{s}_1, \mathbf{s}_2$ such that $\|\!|Z|\!\| = \|\Phi\|_R = \|\Phi\|_{2,1}$

1: $\mathbf{s}_1 = \mathbf{1}$
2: $\mathbf{s}_2 = \|\!|Z|\!\|\boldsymbol{\mu}$
3: **return** $(\mathbf{s}_1, \mathbf{s}_2)$

Algorithm 6 Rescaling for $\|\Phi\|_R = \|\Phi\|_F^2$, $\|C_{:i}\|_C = \max\left(\left\|{C^{(1)}}_{:i}\right\|_2^2, \left\|{C^{(2)}}_{:i}\right\|_2^2\right)$

Input: $Z, \boldsymbol{\mu}$

Output: $\mathbf{s}, \mathbf{s}_2$ such that $\|\!|Z|\!\| = \frac{1}{2}\left\|\text{diag}(\mathbf{s}_2)\Phi\right\|_F^2 + \frac{1}{2}\sum_{i=1}^k \max\left(\left\|\mathbf{c}^{(1)}{}_i \mathbf{s}_i^{(1)}\right\|_2, \left\|\mathbf{c}^{(2)}{}_i \mathbf{s}_i^{(2)}\right\|_2\right)$

1: **for** $i = 1, \ldots, k$ **do**
2: $\quad \mathbf{s}_i^{(1)} = \sqrt{\|\!|Z|\!\|\boldsymbol{\mu}_i} / \left\|\mathbf{c}^{(1)}{}_i\right\|_2^2$
3: $\quad \mathbf{s}_i^{(2)} = \sqrt{\|\!|Z|\!\|\boldsymbol{\mu}_i} / \left\|\mathbf{c}^{(2)}{}_i\right\|_2^2$
4: $\quad \mathbf{s}_2 = \sqrt{\|\!|Z|\!\|\boldsymbol{\mu}}$
5: **end for**
6: **return** $(\mathbf{s}, \mathbf{s}_2)$

convex in the factors. The main difficulty is in finding the induced norm on the joint variable, $Z = C\Phi$, given the chosen regularizers and constraints sets on C and Φ. This chapter presented five cases with computationally efficient induced norms. These cases are used for developing algorithms for several problems in the next three chapters, with an empirical investigation for each setting. A great opportunity remains, however, in finding convex, efficiently computable reformulations for more general settings that encompass clustering and mixture model problems.

This work has focused on convex reformulations to solve regularized factor models globally; there are, however, other approaches to obtaining global solutions. Recently, there has been several theoretical and empirical demonstrations that alternating descent for certain biconvex problems results in global solutions (Journée et al., 2010; Jain et al., 2013; Jain and Dhillon, 2013; Agarwal et al., 2013; Mirzazadeh et al., 2014). These works make the use of regularized factor models more promising, suggesting that a broader class of regularized factorization can be solved globally. I discuss other potential computational advances for regularized factor models in Section 8.1.2.

Chapter 5

Subspace learning and sparse coding using regularized factor models

Data representations are fundamental to machine learning. Expressing complex data objects, such as documents or images, as feature vectors—e.g. as bags of words, vectors of Fourier or wavelet coefficients, or indicators of nearest prototypes—can reveal important structure in a data collection, as well as in individual data items. Feature representations do not only facilitate understanding, they enable subsequent learning. For any particular application, however, often one does not know which representation to use.

Automatically *discovering* useful features from data has been a long standing goal of machine learning research. Current feature discovery methods have already proved useful in many areas of data analysis, including text, image, and biological data processing. These methods differ primarily in the properties sought in any new data representation. Some approaches seek a low dimensional representation, such as principal components analysis (PCA) and modern variants (Van der Maaten and Hinton, 2008; Weinberger and Saul, 2006). Others seek a representation where features behave independently, such as independent components analysis (ICA) (Comon, 1994); or where the new feature vectors are sparse, such as sparse coding or vector quantization (Olshausen and Field, 1997). Still others seek a representation that captures higher level, abstract features of the data that are invariant to low level transformations, such as in deep learning (Hinton, 2007).

In each case, one key issue is whether an optimal feature representation can be recovered efficiently. The lack of an optimal feature discovery method can hamper the practical applicability of a principle—relegating its use to an artform. Globally solvable criteria such as PCA, by contrast, enjoy widespread use despite the numerous shortcomings, arguably because users need not understand the workings of any solver—it is sufficient to understand the principle being optimized. Recently, the development of sparsity inducing regularizers has made great inroads in achieving

globally solvable forms of training. Indeed, convex reformulations have recently had a significant impact on many areas of machine learning, including multitask learning (Argyriou et al., 2008), collaborative filtering (Candes and Recht, 2009; Srebro et al., 2004), and nonlinear dimensionality reduction (Weinberger and Saul, 2006).

In this chapter, I discuss how the advances in convex formulations of regularized factor models can be used to solve three representation learning problems: sparse coding, single-view subspace learning and multi-view subspace learning. In each section, an objective is defined for each problem and a subsequent convex reformulation given. The chapter concludes with an empirical investigation into the advantages of a convex formulation of multi-view subspace learning, both on synthetic data and real image data.

Contributions:

Contribution 1. A clear demonstration that sparse coding using the typical ℓ_1 regularizer has a convex reformulation approach, but that the solution is a trivial form of vector quantization, both for unsupervised and semisupervised learning (Zhang et al., 2011). This result shows that, contrary to popular opinion, ℓ_1 regularization does not lead to over-complete bases and suggests that, unless the rank of the representation is fixed smaller than the input dimension, the ℓ_1 relaxation from ℓ_0 is inadequate. This result was also noted in (Bach et al., 2008), but only for unsupervised learning.

Contribution 2. First global two-view subspace learning algorithm for general convex reconstruction losses (White et al., 2012). In particular, we also provide important efficiency improvements for both the convex optimization and for the recovery to make the two-view subspace learning algorithm more practical.

Contribution 3. An empirical comparison of the convex multi-view subspace learning algorithm to single-view subspace learning and to local solutions for multi-view subspace learning on a variety of datasets (White et al., 2012). This empirical investigation indicates both the usefulness of explicitly treating multiple views separately (rather than concatenated as a single view) and the importance of formulations that obtain global solutions for representation learning.

5.1 Convex sparse coding

In sparse coding, the goal is to learn a sparse representation $\Phi \in \mathbb{R}^{k \times T}$ for input data matrix $X \in \mathbb{R}^{n \times T}$, with a dictionary C such that $X \approx C\Phi$. This goal was introduced (Olshausen and Field, 1997) based on a neural justification that images are represented by a small number of active

code elements; since then, sparse coding has seen widespread use in signal processing and image processing.

The desired regularizer for this problem is the ℓ_0 regularizer, which counts the number of non-zero entries in a vector. Unfortunately, this regularizer is not convex. Instead, a convex proxy that is typically used instead is $\|\Phi\|_{1,1}$, which encourages entry-wise sparsity in Φ (Mairal et al., 2008; Lee et al., 2009; Jenatton et al., 2010). I further justify this choice in Figure 5.1(b).

We can also evaluate this regularizer choice in terms of the prior chosen on Φ. This regularizer corresponds to independent zero-mean Laplace distributions on each entry with diversity parameter $= 1/\alpha$. This Laplace distribution is highly peaked around zero, with heavy tails, giving a prior that favours zero-valued entries in Φ:

$$\begin{aligned}
-\log p(\Phi) &= \sum_{i=1}^{k}\sum_{t=1}^{T} -\log p_{\text{Laplace}}(\Phi_{i,t}|\mu=0, b=1/\alpha) \\
&= -\sum_{i=1}^{k}\sum_{t=1}^{T} \log\left(\frac{1}{2b}\exp\left(\frac{-|\Phi_{i,t}|}{b}\right)\right) \\
&= \sum_{i=1}^{k}\sum_{t=1}^{T} \log(2/\alpha) + \alpha|\Phi_{i,t}|.
\end{aligned}$$

Since the constant is ignored in the optimization, we get $R_{\Phi,k}(\Phi) = \alpha\sum_{i=1}^{k}\sum_{t=1}^{T}|\Phi_{i,t}| = \alpha\|\Phi\|_{1,1}$.

There is rarely any prior information about C, so one might assume that a uniform (non-informative) prior is acceptable. Notice, however, that the factorization $Z = C\Phi$ is invariant to reciprocal rescalings of C and Φ. To avoid degeneracy, therefore, we constrain each column $C_{:i}$ to the bounded closed convex set $\mathcal{C}_q = \{\mathbf{c} \ : \|\mathbf{c}\|_q \leq 1\}$, giving $C \in \mathcal{C}^k$.

The resulting sparse coding problem is

$$\min_{C\in\mathcal{C}_q^k} \min_{\Phi} \ L\left(C\Phi \ ; \ X\right) + \alpha\|\Phi\|_{1,1}.$$

where approximation error is measured with any loss function $L(Z;X)$ that is convex in its first argument (such as the Bregman divergences described in Section 2.2). From Theorem 5, we know that this can be solved globally and efficiently if we relax the rank constraint and allow the optimization to chose rank k. This results in the following convex optimization

$$\min_{C\in\mathcal{C}_q^\infty} \min_{\Phi} \ L\left(C\Phi \ ; \ X\right) + \alpha\|\Phi\|_{1,1} = \min_{Z} \ L(Z;X) + \alpha\|Z'\|_{q,1}.$$

The optimal dictionary and sparse representation can be recovered from the optimal Z, according to Proposition 8 as

$$\begin{aligned}
C &= [Z_{:,1}/\|Z_{:,1}\|_q, \ldots, Z_{:,T}/\|Z_{:,T}\|_q] \\
\Phi &= \text{diag}(\|Z_{:,1}\|_q, \ldots, \|Z_{:,T}\|_q).
\end{aligned}$$

Given a test point $\mathbf{x}$, one can recover $\boldsymbol{\phi} = \arg\min_{\phi} \; L\left(C\boldsymbol{\phi}\,;\,\mathbf{x}\right) + \alpha\|\boldsymbol{\phi}\|_1$, yielding a sparse representation in terms of the training observations.

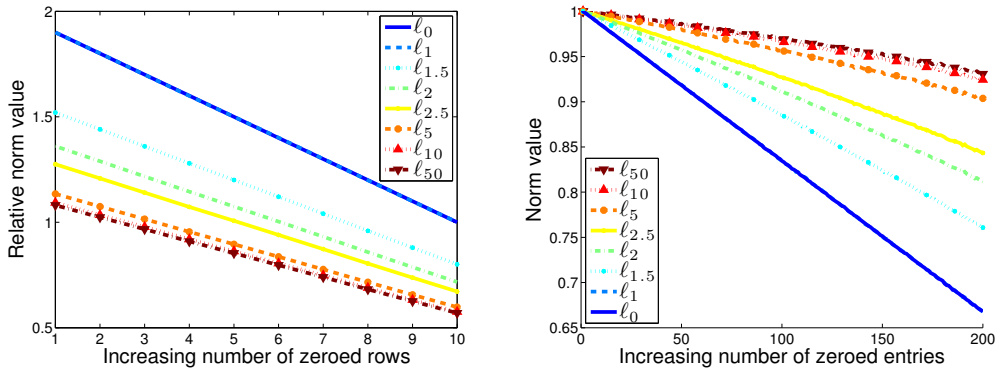
This convex formulation leads to an important insight: the solution is not over-complete, contrary to common intuition. That is, we obtain a simple form of vector quantization that memorizes the (normalized) observations and codes the training data by a scaled indicator vector; an outcome also witnessed by (Bach et al., 2008). This property is an inherent weakness of $\|\cdot\|_{1,1}$ regularization that does not appear to be widely appreciated. One might hope, however, that if the problem was more constrained, the solution would not be an (overfit) memorization of the data and we might regain the useful properties of the ℓ_1 regularizer. In Section 6.2.1, I show that even adding supervised information does not improve generalization; that is, surprisingly, semi-supervised sparse coding does not appear to sufficiently constrain the problem. Another strategy is to use mixed regularizers, such as including a ℓ_2 regularizer on Φ (Bach et al., 2008); unfortunately, already such a simple addition results in a non-convex objective. Understanding how to effectively formulate convex sparse coding, therefore, remains an important, outstanding problem.

5.2 Convex subspace learning

Subspace learning, also typically referred to as dimensionality reduction, is a fundamental problem in representation learning. Re-expressing high-dimensional data in a low dimensional representation has been used to discover important latent information about individual data items, visualize entire data sets to uncover their global organization, and even improve subsequent clustering or supervised learning (Lee and Verleysen, 2010). Classically, convex subspace learning had been achieved using principal components analysis; generalizations to other (robust) losses, however, resulted in non-convex optimizations until the introduction of convex regularizers to impose the low-rank structure, described below.

To find a lower dimensional representation, Φ, such that $\operatorname{rank}(\Phi) = k < n$, we need a prior on Φ that reduces the number of rows of Φ. A widely used regularizer that is believed to encourage entire rows $\Phi_{i:}$ to be zero is the $(2,1)$-block norm, $\|\Phi\|_{2,1}$ (Argyriou et al., 2008). To further justify this choice of regularizer, we look at the underlying prior on Φ and find that it is in fact a Laplacian distribution across samples (White et al., 2015). As discussed before, the Laplacian distribution is peaked at zero, with heavy tails, encouraging zero entries; intuitively, therefore, it makes sense that the $(2,1)$-block norm encourages the entire row to equal zero. First, the connection to Laplacian priors is described; then a simple experiment is included to show that the $(2,1)$-block norm does in fact push entire rows to zero.

There are several multivariate extensions of Laplace distributions; we choose a multivariate



(a) Subspace properties

(b) Sparsity properties

Figure 5.1: Values for various norms on matrices with subspace or sparsity properties. In both graphs, the norm values are scaled to between 0 and 1, so that the trend can be compared. The matrices are generated with independent entries in $\mathcal{N}(0, I)$. **(a)** The values are averaged over 1000 random matrices in $\mathbb{R}^{50\times 30}$ with an increasing number of zeroed rows. To compare the subspace properties of the norms, the norm value is compared to the value on randomly zeroing the same number of entries, rather than rows. The ℓ_0 and ℓ_1 are both above 1 in all cases, having a lower value for zeroing sparsely rather than zeroing an entire row. The ℓ_2 and ℓ_{10} norm both are lower than 1 after the first 12 and 4 rows are zeroed, respectively. When the relative value is lower than 1, the norm value is lower for matrices where the entire row is zeroed rather than just an equivalent number of entries sparsely zeroed. This result suggests that as p gets larger, this property is further enforced. The ℓ_2 norm, in fact, appears to have mixed properties: it prefers sparsity for more dense matrices, and zeroing rows for less dense matrices. **(b)** The values are averaged over 1000 random matrices with an increasing number of zeroed entries, rather than entire rows.

Laplace, parametrized by a mean, $\boldsymbol{\mu}_i$, and scatter matrix Σ_i, with the convenient pdf (Arslan, 2010):

$$p_L(\Phi_{i,:}|\boldsymbol{\mu}_i, \Sigma_i) = \frac{|\Sigma_i|^{-1/2}}{2^T \pi^{\frac{T-1}{2}} \Gamma(\frac{T+1}{2})} e^{-\sqrt{(\Phi_{i,:}-\boldsymbol{\mu}_i)\Sigma_i^{-1}(\Phi_{i,:}-\boldsymbol{\mu}_i)'}}.$$

Assuming $\boldsymbol{\mu} = \mathbf{0}$ and $\Sigma = I$, we get the following negative log-likelihood of the Laplace prior *across samples*

$$\begin{aligned}
&-\log p_L(\Phi_{i,:}|\boldsymbol{\mu}_i = \mathbf{0}, \Sigma_i = I) = \\
&\quad \frac{1}{2}\log\left(|\Sigma_i|\right) + T\log(2) + \frac{T-1}{2}\log(\pi) + \log\Gamma\left(\frac{T+1}{2}\right) + \sqrt{(\Phi_{i,:}-\boldsymbol{\mu}_i)\Sigma_i^{-1}(\Phi_{i,:}-\boldsymbol{\mu}_i)'} \\
&\implies \min_{\Phi} \sum_{i=1}^{k} -\log p_L(\Phi_{i,:}|\boldsymbol{\mu}_i = \mathbf{0}, \Sigma_i = I) = \min_{\Phi} \sum_{i=1}^{k} \sqrt{\Phi_{i,:}\Phi_{i,:}'} = \min_{\Phi} \|\Phi\|_{2,1}.
\end{aligned}$$

To test the appropriateness of the $(2,1)$-block norm we examine the norm values of random Φ matrices, with increasing number of zero rows in Figure 5.1(a) and increasing number of sparse entries in Figure 5.1(b). For Figure 5.1(a), if the y-value is less than 1, then the norm is smaller if entire rows are zeroed, as opposed to zeroing the same number of entries randomly; see the caption for a more detailed description of the experiment. We can see that many of the norms do not

immediately prefer zeroing entire rows; however, after a threshold of zeroed rows is reached, then the preference switches. The larger the p, the more quickly the preference to zero rows is reached. The ℓ_0 and ℓ_1 norms, however, never fall below 1. Therefore, we can see that to obtain subspace learning, selecting $p \geq 2$ is preferable. The ℓ_2 norm is often preferable since it already exhibits these desirable properties, but has better numerical stability. For sparsity, we can see that the trend for ℓ_1 is equivalent to ℓ_0; for $p > 1$, however, the preference for sparsity decreases with increasing p.

As we can see, therefore, an appropriate loss for subspace learning is

$$\min_{C \in \mathcal{C}_2^k} \min_{\Phi} L(C\Phi ; X) + \alpha \|\Phi\|_{2,1}$$

where, as with sparse coding, the columns of $C_{:i} \in \mathcal{C}_2 = \{\mathbf{c} : \|\mathbf{c}\|_2 \leq 1\}$ must be constrained to some set to avoid degeneracy. Since the $(2,1)$-block norm zeros out rows, we do not have to fix k *a priori*, allowing the optimization to instead impute the rank. Now we can apply Theorem 3 to get a convex reformulation

$$\min_{C \in \mathcal{C}_2^\infty} \min_{\Phi} L(C\Phi ; X) + \alpha \|\Phi\|_{2,1} = \min_{Z} L(Z ; X) + \alpha \|Z\|_{tr}.$$

From Proposition 8, given optimal Z with singular value decomposition $Z = U\Sigma V'$, we can recover globally optimal $C = U$ and $\Phi = \Sigma V'$. The solution satisfies $\operatorname{rank}(B) = \operatorname{rank}(\Phi) = \operatorname{rank}(Z)$; thus, even though we allowed $C \in \mathcal{C}_2^\infty$, the dimension is implicitly and efficiently controlled by reducing the rank of Z via trace norm regularization. Though we arrived at this regularizer differently, the trace norm has already widely been used as a convex relaxation of rank in low rank matrix recovery (Candes and Recht, 2009; Salakhutdinov and Srebro, 2010), because $\|X\|_{tr}$ is the convex envelope of $\operatorname{rank}(X)$ over the set $\{X : \|X\|_{sp} \leq 1\}$ (Recht et al., 2010).

For new data points, X_{new}, we can easily find its lower-dimensional representation $Z_{\text{new}} = C\Phi_{\text{new}}$ using the learned basis weights, C

$$\min_{\Phi_{\text{new}}} L(C\Phi_{\text{new}} ; X_{\text{new}}) + \alpha \|\Phi_{\text{new}}\|_{2,1}$$

which is convex in Φ_{new} for a fixed C. Therefore, this view of subspace learning enables both a convex reformulation as well as a simple out-of-sample approach.[1]

Interestingly, we can also extend this convex subspace formulation to remove sparse noise in X, generalizing the robust subspace learning formulations of (Candes and Recht, 2009; Xu et al.,

[1] Note that the solutions are different for computing the m samples jointly in $\Phi_{\text{new}} = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_m]$ versus optimizing each $\boldsymbol{\phi}_t$ independently. This difference results from the fact that the $(2,1)$-block norm couples instances across time. More practically, this coupling is evident in the gradient computation, where $\partial \|\Phi\|_{2,1} / \partial \Phi_{jt} = (\sum_{i=1}^k \Phi_{it})^{-1/2} \Phi_{jt}$. For other regularizers, such as the Frobenius norm regularizer or $(1,1)$-block norm regularizer, optimizing independently versus jointly is equivalent.

2010). In Corollary 4, the ℓ_1 regularizer is used directly on a sparse noise matrix, S, and so does not result in the same vector quantization properties as the sparse coding matrix factorization in the previous section.

Corollary 4 (Convex robust subspace learning)**.**

$$\begin{aligned}\min_{C \in \mathcal{C}_2^\infty} \min_{\Phi} \min_{S} L\left(C\Phi + S\ ;\ X\right) + \alpha\|\Phi\|_{2,1} + \beta\|S\|_{1,1} \\ = \min_{Z} \min_{S} L\left(Z + S\ ;\ X\right) + \alpha\|Z\|_{tr} + \beta\|S\|_{1,1}.\end{aligned}$$

This objective can be globally optimized by alternating between Z and S, because they interact additively rather than multiplicatively in the loss. Therefore, the objective is jointly convex in Z and S. Using the singular value decomposition, $Z = U\Sigma V'$, we can again see that the recovery is $C = U$ and $\Phi = U\Sigma$ since $\|\Phi\|_{2,1} = \mathrm{tr}(\Sigma) = \|Z\|_{tr}$, regardless of the shift by S.

5.3 Convex multi-view subspace learning

Modern data is increasingly complex, with increasing size and heterogeneity. For example, multimedia data analysis considers data objects (e.g. documents or webpages) described by related text, image, video, and audio components. *Multi-view learning* focuses on the analysis of such multi-modal data by exploiting its implicit conditional independence structure. For example, given multiple camera views of a single object, the particular idiosyncrasies of each camera are generally independent, hence the images they capture will be *conditionally independent* given the scene. Similarly, the idiosyncrasies of text and images are generally conditionally independent given a topic. The goal of multi-view learning, therefore, is to use known conditional independence structure to improve the quality of learning results. Figure 5.2 indicates the different independence assumptions between single-view learning and multi-view learning. To formally explain this structure, Figure 5.3 contains the well-known d-separation rules for graphical models. The tail-to-tail rule in Figure 5.3(b) indicates that, for the multi-view model in Figure 5.2(b), any two views are conditionally independent given the latent node. In addition to this theoretical conditional independence structure, empirically respecting the split between views has been shown to improve performance (Nigam and Ghani, 2000).

Classically, *multi-view subspace learning* has been achieved by an application of canonical correlation analysis (CCA) (Hardoon et al., 2004; De Bie et al., 2005). Recall from Proposition 6 in Section 3.2.2, that CCA is actually equivalent to the regularized factorization

$$(C, \Phi) = \arg\min_{C,\Phi} \left\| \begin{bmatrix} (X_1 X_1{}')^{-1/2} X_1 \\ (X_2 X_2{}')^{-1/2} X_2 \end{bmatrix} - C\Phi \right\|_F^2$$

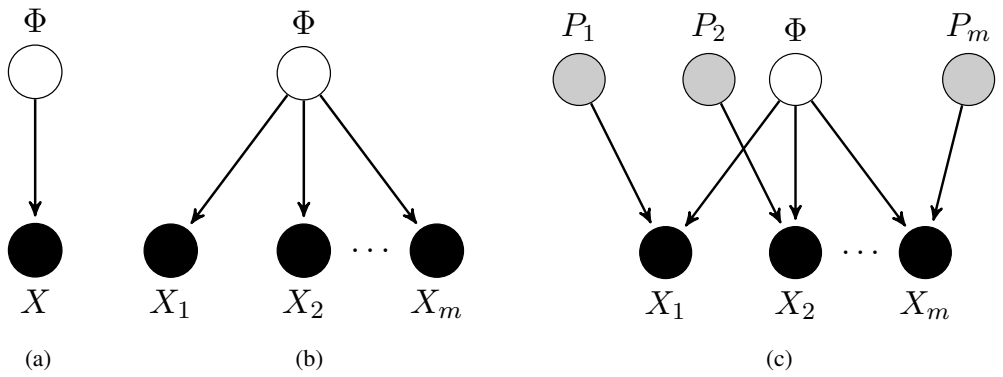
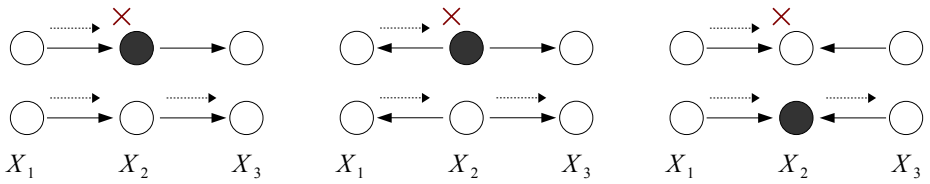


Figure 5.2: Different dependency relationships between views. Dark grey means the variable is observed.
(a) Single-view Representation: Each view has its own latent representation, such as is typical in standard single-view subspace learning and sparse coding.
(b) Multi-view Shared Latent Representation: This structure indicates a shared latent representation that makes the views $X_1, \ldots, X_m$ conditionally independent. In practice, this structure typically underlies algorithms that attempt to recover X_i using the factorization $C_i\Phi$.
(c) Multi-view with Private and Shared Information: In addition to the conditional independence structure, the explicit private information is used to describe certain algorithms that explicitly learn P_i or parameters for P_i. In Figure (b), these quantities are implicitly the remaining noise after obtaining $C_i\Phi$. Probabilistic PCA, on the other hand, explicitly learns the parameter σ_i for $P_i \sim \mathcal{N}(\mathbf{0}, \sigma_i I)$.

with the solution satisfying $C^{(1)\prime}C^{(1)} = C^{(2)\prime}C^{(2)} = I$. One can see, therefore, that this formulation respects the conditional independence of the separate views: given a latent representation $\boldsymbol{\phi}_j$, the reconstruction losses on the two views cannot influence each other, since the reconstruction models $C^{(1)}$ and $C^{(2)}$ are *individually* constrained. By contrast, in single-view subspace learning, $X = [X_1; X_2]$ are not normalized (shown in Proposition 2 for PCA), and so the concatenated $C^{(1)}$ and $C^{(2)}$ are constrained as a whole. Consequently, $C^{(1)}$ and $C^{(2)}$ must then compete against each other to acquire magnitude to explain their respective "views" given $\boldsymbol{\phi}_j$ (i.e. conditional independence is not enforced). Such sharing can be detrimental if the two views really are conditionally independent given $\boldsymbol{\phi}_j$.

Despite its elegance, a key limitation of CCA is its restriction to a squared loss under a particular normalization. Many successes have been achieved in using CCA to recover meaningful latent representations in a multi-view setting (Dhillon et al., 2011; Lampert and Kromer, 2010; Sigal et al., 2009). Such work has also been extended to probabilistic (Bach and Jordan, 2006) and sparse formulations (Archambeau and Bach, 2008). CCA-based approaches, however, only admit efficient global solutions when using the squared-error loss (i.e. Gaussian models), while extensions to robust models have had to settle for approximate solutions (Viinikanoja et al., 2010).

In this section, I derive a convex formulation for generalized multi-view subspace learning for two views, $X_1 \in \mathbb{R}^{n_1 \times T}$ and $X_2 \in \mathbb{R}^{n_2 \times T}$ (White et al., 2012). To the best of my knowledge, this



(a) Head-to-Tail (b) Tail-to-Tail (c) Head-to-Head

Figure 5.3: The d-separation rules for causal graphs (Geiger et al., 1990). **(a)** If X_2 is **observed** (i.e. given), then X_3 is **independent** of X_1 (i.e. d-separated). If X_2 is **not observed**, then X_3 is **dependent** on X_1 (i.e. d-connected). **(b)** Same as Head-to-Tail. **(c)** If X_2 is **not observed** (i.e. not given), then X_3 is **independent** of X_1 (i.e. d-separated). If X_2 is **observed**, then X_3 is **dependent** on X_1 (i.e. d-connected). Note that for (c), X_2 can also be indirectly observed through a descendant. If there was a node, A, such that $X_2 \to A$, and A was observed, then X_1 and X_3 would be dependent.

is the first global algorithm for multi-view subspace learning. An extension to more than two views remains future work, though recent results suggest this extension may require a relaxation (Zhang et al., 2012).

Similarly to single-view subspace learning, the relaxed-rank optimization for two-view subspace learning is

$$\min_{C^{(1)},C^{(2)},\Phi} L\left(\begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix} \Phi\,;\, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha\|\Phi\|_{2,1}, \quad \text{s.t., for } C = \begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix}, C_{:i} \in \mathcal{C} \text{ for all } i \tag{5.1}$$

where

$$\mathcal{C} := \left\{ \begin{bmatrix} \mathbf{c}^{(1)} \\ \mathbf{c}^{(2)} \end{bmatrix} : \|\mathbf{c}^{(1)}\|_2 \le \beta_1, \|\mathbf{c}^{(2)}\|_2 \le \beta_2 \right\}. \tag{5.2}$$

Recall from Section 5.2 that the $(2,1)$-block norm regularizer $\|\Phi\|_{2,1} = \sum_i \|\Phi_{i,:}\|_2$ encourages *rows* of Φ to be sparse. Again, C must be constrained; otherwise $\|\Phi\|_{2,1}$ can be pushed arbitrarily close to zero simply by re-scaling Φ/s and Cs ($s > 0$) while preserving the same loss. In this case, however, the two dictionaries $C^{(1)}$ and $C^{(2)}$ must be constrained separately in order to maintain the graphical model structure in Figure 5.2.

We can again exploit the general convex reformulation techniques developed for regularized factor models to get the first convex two-view subspace learning optimization for general convex losses. Using Theorem 7, we obtain the equivalent convex optimization

$$(5.1) = \min_{Z} L\left(Z\,;\, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha \max_{0\le\eta\le 1} \left\|E_\eta^{-1} Z\right\|_{tr}$$

where

$$E_\eta := \begin{bmatrix} \beta_1/\sqrt{\eta}\, I_{n_1} & 0 \\ 0 & \beta_2/\sqrt{1-\eta}\, I_{n_2} \end{bmatrix}.$$

The loss, L, can be viewed as a sum of separate matching losses: $L\left(C\Phi\ ;\ Z\right) = D_{F_1^*}(C^{(1)}\Phi; X_1) + D_{F_2^*}(C^{(2)}\Phi; X_2)$. The understanding of the relationship between transfers/losses and distributions described in Section 2.2, therefore, naturally extends to the multi-view setting.

This formulation can be further improved for an efficient implementation. Though 5.1 is convex, naively applying a gradient descent approach and computing the subgradients for η and Z for the trace norm is computationally inefficient. First, since η is not involved in the reconstruction loss, L, we can move the maximum over η left:

$$(5.1) = \min_{Z} \max_{0\leq\eta\leq 1} L\left(Z\ ;\ \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha \left\|E_\eta^{-1} Z\right\|_{tr}.$$

Then, we can swap the max and min due to strong convexity (Rockafellar, 1970, Cor. 37.3.2),

$$(5.1) = \max_{0\leq\eta\leq 1} \min_{Z} L\left(Z\ ;\ \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha \left\|E_\eta^{-1} Z\right\|_{tr}$$

to get an equivalent concave-convex maxi-min problem. Next, the change of variables, $Q = E_\eta^{-1} Z$, leads to an equivalent but computationally more convenient formulation:

$$(5.1) \ = \ \max_{0\leq\eta\leq 1} \min_{Q} L\left(E_\eta Q\ ;\ \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha \left\|Q\right\|_{tr}. \tag{5.3}$$

The transformation does not affect the concavity of the problem with respect to η (see Appendix E). The crucial improvement from this slight change is that the inner minimization in Q is a standard trace-norm-regularized loss minimization problem, which has been extensively studied in the matrix completion literature (Ma et al., 2011; Cai et al., 2010; Zhang et al., 2012). The concave outer maximization is defined over a *scalar* variable η, hence simple line search can be used to solve the problem, normally requiring at most a dozen evaluations to achieve a small tolerance. Thus we have achieved an efficient, globally solvable formulation for two-view subspace learning that respects conditional independence of the separate views.

The training procedure consists of two stages: first, solve (5.3) to recover η and Q, which allows $Z = E_\eta Q$ to be computed; then, recover the optimal factors Φ and C (*i.e.* $C^{(1)}$ and $C^{(2)}$) from Z. Chapter 4 provides a simple and efficient boosting procedure for recovering C and Φ from Z, in Algorithm 2 with oracle in Algorithm 3. Pseudocode summarizing this procedure is given in Algorithm 7. Notice that when $\beta_1 = \beta_2 = \beta$ and $\eta = 0.5$, equally weighting the two views, (5.3) reduces to single-view subspace learning:

$$\begin{aligned}\min_{Q} L\left(E_\eta Q\ ;\ \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha \left\|Q\right\|_{tr} &= \min_{Q} \frac{\beta}{\sqrt{0.5}} L\left(Q\ ;\ \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha \left\|Q\right\|_{tr} \\ &= \min_{Z} L\left(Z\ ;\ \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \frac{\alpha\sqrt{0.5}}{\beta} \left\|Z\right\|_{tr}.\end{aligned}$$

Algorithm 7 Convex two-view subspace learning

Input: $X_1, X_2, \alpha, \beta_1, \beta_2$

Output: Z, C, Φ

Initialize $\eta = 0.5$ // or another random value $\eta \in (0, 1)$

Initialize $Q = E_\eta \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} \sqrt{\eta}/\beta_1 \, X_1 \\ \sqrt{1-\eta}/\beta_2 \, X_2 \end{bmatrix}$

// Define the inner function we want to minimize

[f,g] = InnerLoss(Q, η):

    $[U, \Sigma, V] = \text{svd}(Q)$

    $v_1 = \beta_1/\sqrt{\eta}$

    $v_2 = \beta_2/\sqrt{1-\eta}$

    $\text{f} = L\left(\begin{bmatrix} v_1 Q^{(1)} \\ v_2 Q^{(2)} \end{bmatrix}; \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha \, \text{sum}(\text{diag}(\Sigma))$

    $\text{g} = \nabla L_1(v_1 Q^{(1)}, X_1) v_1 + \nabla L_2(v_2 Q^{(2)}) v_2 + \alpha U V'$

// Define the line search function we want to minimize

[f,g] = OuterLineSearch(η):

    // Apply your favourite non-smooth optimizer to find the inner solution

    [Q,f] = lbfgs(InnerLoss($\cdot, \eta$), Q)

    $v_1 = \beta_1/\sqrt{\eta}$

    $v_2 = \beta_2/\sqrt{1-\eta}$

    $\nabla G_1 = L_1(v_1 Q^{(1)}, X_1)$

    $\nabla G_2 = L_2(v_2 Q^{(2)}, X_2)$

    $dv_1 = -\beta_1 \eta^{-3/2}/2$

    $dv_2 = \beta_2 (1-\eta)^{-3/2}/2$

    $\text{g} = dv_1 \text{sum}(\text{sum}(Q^{(1)} \circ G_1)) + dv_2 \text{sum}(\text{sum}(Q^{(2)} \circ G_2))$

// Apply your favourite optimizer to the line search; save resulting minimum Q

[η, Q] = lbfgs(OuterLineSearch, η)

$Z = E_\eta Q = \begin{bmatrix} \beta_1/\sqrt{\eta} \, X_1 \\ \beta_2/\sqrt{1-\eta} \, X_2 \end{bmatrix}$

// Now recover the optimal C and Φ using boosting

(C, Φ) = Algorithm 2(Z, $\|E_\eta \cdot\|_{tr}$, Oracle Alg 3, Rescaling Alg 5)

return Z, C, Φ

5.4 Experimental results for convex multi-view subspace learning

This section describes an experimental evaluation of the multi-view latent representation learning algorithm (White et al., 2012). I first describe the algorithms that were compared and then present results in two empirical settings: synthetic data (to compare optimization quality and runtime) and image denoising on a face image dataset.

Comparative Training Procedures

Below we compare the proposed global learning method, **Multi-view Subspace Learning (MSL)**, against two benchmark competitors.

Local Multi-view Subspace Learning (LSL) An obvious competitor is to solve (5.1) by alternating descent over the variables: optimize Φ with $C^{(1)}$ and $C^{(2)}$ fixed, optimize $C^{(1)}$ with $C^{(2)}$ and

Φ fixed, etc. This is the computational strategy employed by Quadrianto and Lampert (2011); Jia et al. (2010). Since $C^{(1)}$ and $C^{(2)}$ are both constrained and Φ is regularized by the (2,1)-block norm which is not smooth, we optimized them using the proximal gradient method (Beck and Teboulle, 2009).

Single-view Subspace Learning (SSL) Single-view learning can be cast as a relaxation of (5.1), where the columns of $C = \begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix}$ are normalized *as a whole*, rather than individually for $C^{(1)}$ and $C^{(2)}$:

$$\begin{aligned} \min_{\{\Phi, C : \|C_{:,i}\|_2 \leq \sqrt{{\beta_1}^2 + {\beta_2}^2}\}} & L\left(C\Phi\,;\, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha\|\Phi\|_{2,1} \\ &= \min_{\{\Phi, C : \|C_{:,i}\|_2 \leq 1\}} L\left(C\Phi\,;\, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha({\beta_1}^2 + {\beta_2}^2)^{-\frac{1}{2}}\|\Phi\|_{2,1} \\ &= \min_{Z} L\left(Z\,;\, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) + \alpha({\beta_1}^2 + {\beta_2}^2)^{-\frac{1}{2}}\|Z\|_{\mathrm{tr}}. \end{aligned} \tag{5.4}$$

Equation (5.4) follows from the analysis in the previous section on single-view subspace learning and matches the formulation given in (Candes et al., 2011). To solve (5.4) when α is large, we used a variant of the boosting algorithm (Zhang et al., 2012), due to its effectiveness when the solution has low rank. When α is small, making the rank regularizer not as influential in the optimization, we switch to the alternating direction augmented Lagrangian method (ADAL) (Goldfarb et al., 2010) which does not include the trace norm in all iterations. This hybrid choice of solver is also applied to the optimization of Q in (5.3) for MSL. For the single-view approximation, once an optimal Z is achieved, the corresponding C and Φ can be recovered by a singular value decomposition: for $Z = U\Sigma V'$, set $C = ({\beta_1}^2 + {\beta_2}^2)^{\frac{1}{2}}U$ and $\Phi = ({\beta_1}^2 + {\beta_2}^2)^{-\frac{1}{2}}\Sigma V'$ (see Proposition 8).

Model specification Since all datasets have sparse noise, $L_{1,1}$ loss is an appropriate choice for L:

$$L\left(\begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix}\Phi\,;\, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\right) = \|C^{(1)}\Phi - X_1\|_{1,1} + \|C^{(2)}\Phi - X_2\|_{1,1}.$$

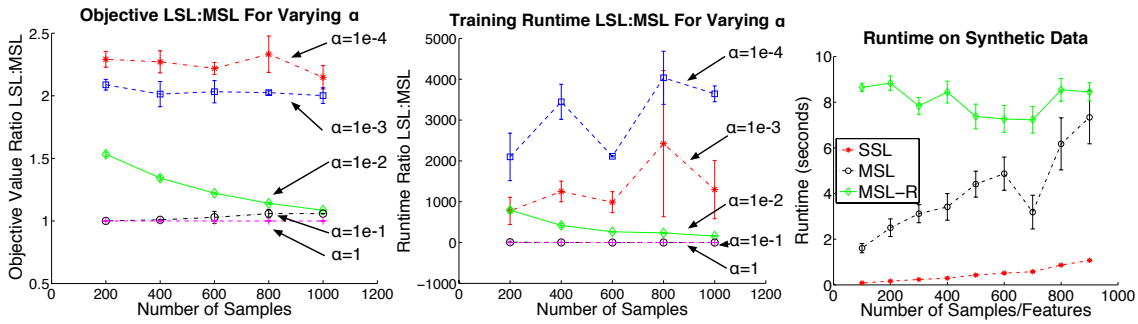
For computational reasons, we used a smoothed version of the $L_{1,1}$ loss (Goldfarb et al., 2010), which uses ℓ_2 for matrix entries near zero:

$$L_{1,1,\sigma}(X) = \sum_i \sum_j \begin{cases} |X_{ij}| - \frac{\sigma}{2} & \text{if } |X_{ij}| \geq \sigma \\ X_{ij}^2/(2\sigma) & \text{if } |X_{ij}| < \sigma \end{cases}$$

Comparing optimization quality

We compared the optimization quality and speed of MSL, LSL and SSL on synthetic data.

Dataset: The synthetic dataset is generated as follows. First, we randomly generate a k-by-t_{tr} matrix Φ_{tr} for training, a k-by-t_{te} matrix Φ_{te} for testing, and two basis matrices, $C^{(1)}$ (n-by-k)



(a) Objectives for LSL:MSL (b) Runtimes for LSL:MSL (c) Runtimes for SSL and MSL

Figure 5.4: Comparison between LSL and MSL on synthetic datasets with changing α, $n = m = 20$ and 10 repeats. (a) LSL often gets stuck in local minima, with a significantly higher objective than MSL. (b) For small α, LSL is significantly slower than MSL. They scale similarly with the number of samples (c) Runtimes of SSL and MSL for training and recovery with $\alpha = 10^{-3}$. For growing sample size, $n = m = 20$. MSL-R stands for the recovery algorithm. The recovery time for SSL is almost 0, so it is not included.

and $C^{(2)}$ (m-by-k), by (i.i.d.) sampling from a zero-mean unit-variance Gaussian distribution. The columns of $C^{(1)}$ and $C^{(2)}$ are then normalized to ensure that the Euclidean norm of each is 1. Then we set

$$X_{1\mathrm{tr}} = C^{(1)}\Phi_{\mathrm{tr}}, \quad X_{2\mathrm{tr}} = C^{(2)}\Phi_{\mathrm{tr}}, \quad X_{1\mathrm{te}} = C^{(1)}\Phi_{\mathrm{te}}, \quad X_{2\mathrm{te}} = C^{(2)}\Phi_{\mathrm{te}}.$$

Next, we add noise to these matrices, to obtain $\tilde{X}_{1\mathrm{tr}}$, $\tilde{X}_{2\mathrm{tr}}$, $\tilde{X}_{1\mathrm{te}}$, $\tilde{X}_{2\mathrm{te}}$. Following (Candes et al., 2011), we use sparse non-Gaussian noise: 5% of the matrix entries were selected randomly and replaced with a value drawn uniformly from $[-M, M]$, where M is 5 times the maximal absolute entry of the matrices.

Comparison: We first compare the optimization performance of MSL (global solver) versus LSL (local solver). Figure 5.4(a) indicates that MSL consistently obtains a lower objective value, sometimes by a large margin: more than two times lower for $\alpha = 10^{-4}$ and 10^{-3}. As α increases, the difference shrinks, suggesting that more local minima occur in the higher rank case (a large α increases regularization and decreases the rank of the solution). In the following experiments on denoising image datasets, we will see that the lower optimization quality of LSL and the fact that SSL optimizes a less constrained objective both lead to noticeably worse denoising performance.

Second, we compare the runtimes of the three algorithms. Figure 5.4(b) presents runtimes for LSL and MSL for an increasing number of samples. Again, the runtime of LSL is significantly worse for smaller α, as much as 4000x slower; as α increases, the runtimes become similar. This result is likely due to the fact that for small α, the MSL inner optimization is much faster via the ADAL solver (the slowest part of the optimization), whereas LSL still has to slowly iterate over the

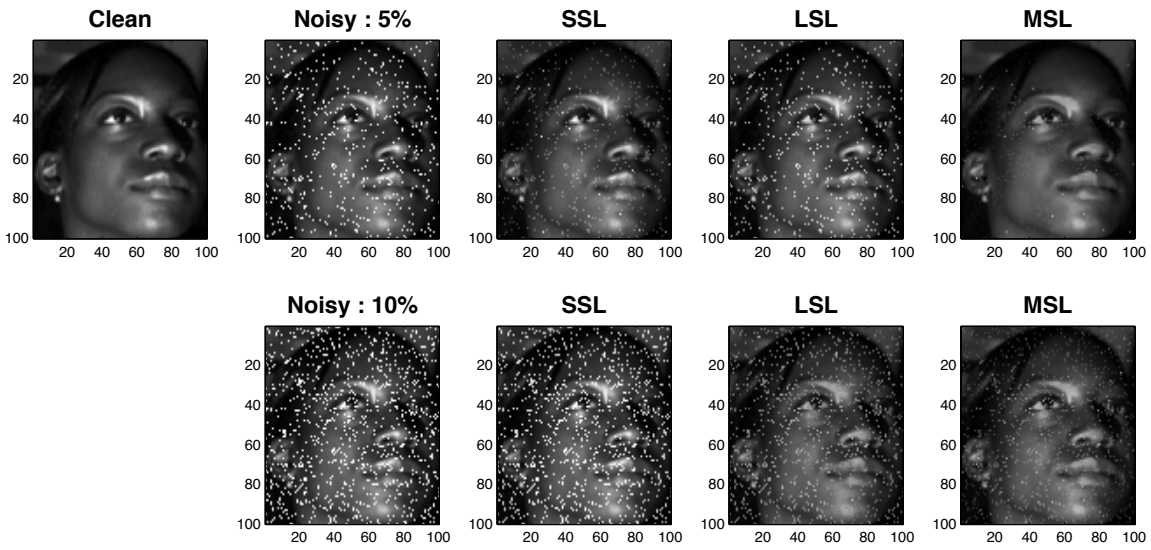


Figure 5.5: Reconstruction of a noisy image with 5% or 10% noise. LSL performs only slightly worse than MSL for larger noise values: a larger regularization parameter is needed for more noise, resulting in fewer local minima (as discussed in Figure 1). Conversely, SSL performs slightly worse than MSL for 5% noise, but as the noise increases, the advantages of the MSL objective are apparent.

three variables. They both appear to scale similarly with respect to the number of samples.

For SSL versus MSL, we expect SSL to be faster than MSL because it is a more straightforward optimization: the SSL objective has the same form as the inner optimization of the MSL objective in (5.3) over Q (with a fixed η). Figure 5.4(c), however, illustrates that this difference is not substantial for increasing sample size. Interestingly, the recovery runtime seems independent of dataset size, and is instead likely proportional to the rank of the data. MSL scales similarly with increasing features as with increasing samples, requiring only about a minute for 1000 features.

Comparing denoising quality

Next we compare the denoising capabilities of the algorithms on a face image dataset.

Datatset: The image dataset is based on the Extended Yale Face Database B (Georghiades et al., 2001). It contains grey level face images of 28 human subjects, each with 9 poses and 64 lighting conditions. To construct the dataset, we set view-one to a fixed lighting (+000E+00) and view-two to a different fixed lighting (+000E+20). We obtain a pair of views by randomly drawing a subject and a pose (under the two fixed lightings). The underlying assumption is that each lighting has its own set of bases ($C^{(1)}$ and $C^{(2)}$) and each (person, pose) pair has the same latent representation for the two lighting conditions. All images are down-sampled to 100-by-100, meaning $n = m = 10^4$. We kept one view (view-one) clean and added pixel errors to the second view (view-two). We randomly set 5% of the pixel values to 1, motivated by the types of noise typical for images, such

as occlusions and loss of pixel information from image transfer. The goal is to enable appropriate reconstruction of a noisy image using another view.

Comparison: We compared the reconstructions, ${Z^{(1)}}_{\text{te}}$ and ${Z^{(2)}}_{\text{te}}$, with the clean data, $X_{1\text{te}}$ and $X_{2\text{te}}$, in terms of the signal-to-noise ratio:

$$SNR({Z^{(1)}}_{\text{te}}, {Z^{(2)}}_{\text{te}}) = \left(\|X_{1\text{te}}\|_F^2 + \|X_{2\text{te}}\|_F^2\right) \Big/ \left(\left\|X_{1\text{te}} - {Z^{(1)}}_{\text{te}}\right\|_F^2 + \left\|X_{2\text{te}} - {Z^{(2)}}_{\text{te}}\right\|_F^2\right).$$

We cross-validated over $\alpha \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 0.5, 1\}$ according to the highest signal-to-noise ratio on the training data. We set $\beta_2 = \beta_1 = 1$ because the data is in the $[0, 1]$ interval.

In Figure 5.5, we can see that MSL outperforms both SSL and LSL on the face image dataset for two noise levels: 5% and 10%. MSL had on average a 10x higher SNR than SSL and significantly different objective values. SSL had higher reconstruction error on the clean view-one (10x higher), lower reconstruction error on the noisy view-two (3x lower) and a higher representation norm (3x higher). The noisy view-two likely skewed the representation, due to the joint rather than separate constraint as in the MSL objective.

5.5 Summary

This chapter illustrated that the previous convex reformulations could be effectively applied to both single-view and multi-view subspace learning. Experimental results indicated that the algorithms developed using convex reformulations outperformed the local alternating algorithms, both in terms of objective values and in computational efficiency. The multi-view structure, encoded by the partitioned constraints and weighted trace norm, was shown to improve denoising of images over the single-view structure, given availability of multiple views of the images.

This chapter explored the idea that problem properties can be nicely captured using regularizers. To this point, most unsupervised learning algorithms encoded different properties using different losses, kernels and instance weights (see the table in Section 3.4). In this chapter, the subspace and sparsification properties of convex ℓ_p norms, from $p = 1$ to $p = 50$ were explored, indicating that ℓ_1 closely matches the sparsity properties of ℓ_0 and that as p increases, the regularizer prefers lower rank Φ. The difference between single-view and multi-view learning was encoded by different choices of regularizer on $Z = C\Phi$. Finally, for least-squares losses, closed form solutions were possible with a hard rank constraint; for general losses, however, the use of a regularizer is key for obtaining convex reformulations. Using regularization to restrict the rank, therefore, enabled convex subspace learning with a robust loss. Overall, from the applications described in this chapter, we saw that using regularization is useful in terms of placing priors on the parameters, encoding properties that may not be related to a known prior and enabling subspace learning with general losses.

Chapter 6

Semi-supervised learning using regularized factor models

The goal of semi-supervised learning is to improve supervised prediction performance by using unlabeled data. Typically, there is much more unlabeled data available than labeled data; it can be unclear, however, how to use this unlabeled data to improve supervised learning. In particular, given the diversity of supervised and unsupervised training principles, it is often unclear how they can best be combined. There remains a proliferation of proposed approaches (Zhu, 2006) with no theoretical guarantees that exploiting unlabeled data will even avoid harm (Ben-David et al., 2008; Li and Zhou, 2011; Nadler et al., 2009). Although some unification can be achieved between supervised and unsupervised learning in a pure probabilistic framework (Bishop, 2006), here too it is not known which unifying principles are appropriate for discriminative models, and a similar diversity of learning principles exists (Smith and Eisner, 2005; Corduneanu and Jaakkola, 2006). The dominant approaches consist of

i) using a supervised loss with an unsupervised loss regularizer, including many graph-based methods (Belkin et al., 2006; Corduneanu and Jaakkola, 2006; Zhou and Schölkopf, 2006);

ii) combining self-supervised training on the unlabeled data with supervised training on the labeled data (Cristianini, 2003; Joachims, 1999);

iii) training a joint probability model generatively (Bishop, 2006; Druck and McCallum, 2010; Nigam et al., 2000);

iv) co-training (Blum and Mitchell, 1998); and

v) using unsupervised training on the entire dataset to learn new features for supervised training on the labeled set, such as sparse features (Lee et al., 2009) or lower-dimensional features (Pereira and Gordon, 2006; Rish et al., 2008).

These dominant approaches can be generally categorized into two types of semi-supervised learning. In the first, which I will call *standard semi-supervised learning*, the goal is to learn weights directly on the given data using both labeled and unlabeled data. Canonical examples of this approach are co-training (Blum and Mitchell, 1998) and transductive support vector machines (Joachims, 1999); from above, they also include approaches (i)-(iv). In the second, which I will call *representation-imputed semi-supervised learning*, a new representation is learned using the unlabeled data. Simultaneously, that "better" representation is used to facilitate learning on the small amount of labeled data. A simple example of this approach is a two-staged learning approach, such as applying PCA to the dataset of concatenated inputs for the labeled and unlabeled datasets, extracting features for the labeled dataset and finally performing supervised learning on these features. This category corresponds to the above work in (v).

In this chapter, we see how using regularized factor models promotes algorithm development for both of these semi-supervised learning approaches. The forward-reverse prediction formalism enables development of a principled standard semi-supervised learning algorithm, where the unlabeled data reduces the variance of the error estimate, indicating provable benefits of the addition of unlabeled data. Interestingly, we currently only have such a guarantee for this first form of semi-supervised learning, facilitated by the fact that the supervised and unsupervised losses are combined as comparable reverse prediction losses. For the second form, where the unlabeled data is used to learn a new representation that is then used for supervised prediction problem, there does not yet appear to be such guarantees because the training principles between the labeled and unlabeled data are quite different (i.e. a reverse loss and a forward loss). The second form of semi-supervised learning, however, is amenable to convex reformulation, whereas the first form does not yet appear to be. Combined, they provide two different semi-supervised learning approaches with two useful, but distinct, guarantees.

Contributions:

Contribution 1. The first semi-supervised learning objective for which we can show that unlabeled data can produce a lower variance estimate of performance, and otherwise does not harm learning performance over strictly using supervised learning. In particular, the objective decomposes into two expectations; the approximation of one of these expectations is improved with more unlabeled data, resulting in a reduced variance estimate. This objective and proof were first developed for least-squares losses (Xu et al., 2009), and later generalized to any Bregman divergence (White and Schuurmans, 2012). This guarantee is possible due to using a reverse loss (i.e., regularized factor model loss) for the supervised learning component.

Contribution 2. An approximation to the principled semi-supervised objective that enables the development of novel non-linear semi-supervised k-means and mixture-model clustering algorithms (White and Schuurmans, 2012). These algorithms extend previous unsupervised linear Bregman clustering algorithms (Banerjee et al., 2005). I present an empirical demonstration that the generality of modeling choices under regularized factor models facilitates novel extensions on widely used algorithms, such as generalizations to normalized cut, and generally enables superior performance.

Contribution 3. A convex representation-imputed semi-supervised learning algorithm (Zhang et al., 2011) using the convex reformulations from Chapter 4. This reformulation is similar to the two-view formulation, with the inputs X as one view and the targets Y as the other view. The only previous convex formulation (Goldberg et al., 2010) uses a transductive dimensionality reduction formulation that 1) does not enable extraction of the representation Φ nor the prediction model and 2) cannot enforce individual constraints on the supervised and unsupervised parts of the model.

Contribution 4. An empirical demonstration that the convex semi-supervised learning algorithm (Zhang et al., 2011) improves upon 1) the previous convex algorithm (Goldberg et al., 2010), due to the importance of the partitioned constraints and 2) alternating and staged solvers for a similar objective.

6.1 Theoretically sound standard semi-supervised learning

Despite their long parallel history, the principles that underly unsupervised learning are often distinct from those underlying supervised learning. The lack of a unification between supervised and unsupervised learning might not be a hindrance if the two tasks are considered separately, but for semi-supervised learning one is forced to consider both together. There has been some theoretical investigation into using unlabeled data to reduce the hypothesis space for supervised learning (Balcan and Blum, 2005); this theoretical understanding, however, can be difficult to generalize. Instead, the literature relies on intuitions like the "cluster assumption" and the "manifold assumption" to provide helpful guidance (Belkin et al., 2006), but these have yet to lead to a general characterization of the potential and limits of semi-supervised learning.

Using regularized factor models, however, the disparity between supervised and unsupervised learning can be overcome to develop a natural principle for semi-supervised learning. Recall from Sections 3.1 and 3.2 that both supervised and unsupervised learning problems can be represented as regularized factor models. Consequently, one can perform semi-supervised learning in a unified way using a reverse loss for both labeled and unlabeled data:

$$\min_{C \in \mathcal{C}} \min_{\Phi \in \mathcal{F}} \; D_{F^\star}(CY_l || \mathbf{f}(X_l))/T_l \;\; + \;\; \mu \, D_{F^\star}(C\Phi || \mathbf{f}(X_u))/T_u. \tag{6.1}$$

Here (X_l, Y_l) and X_u denote the labeled and unlabeled data, T_l and T_u denote the respective number of examples, the parameter μ trades off between the two losses and Φ denotes the labels to impute for the unlabeled data.

Despite the simplicity of the objective in Equation (6.1), it has not apparently been investigated in the literature previously. This could be due to the fact that it has been typical to combine a forward loss on the labeled data and a reverse loss (or regularizer) on the unlabeled data (Belkin et al., 2006; Kulis et al., 2009; Zhou et al., 2004). The use of a reverse loss on the supervised component is a simple proposal, but enables a decomposition into a loss that depends on the input data and a loss that depends on the output data. This resulting decomposition enables variance reduction with the addition of auxiliary unlabelled data. The objective in (6.1) is a close approximation, inspired by this theoretical analysis, but more straightforward to optimize. In the next section, we describe this principled objective and variance reduction argument, showing also that (6.1) is a closely related objective. Section 6.1.2 then presents algorithms to solve Equation (6.1) for semi-supervised classification and regression. This section concludes with an empirical demonstration indicating the advantages of this reverse prediction semi-supervised approach, including improved performance from the diversity of available modeling choices under regularized factor models.

6.1.1 Variance reduction using unlabeled data

When using semi-supervised learning, it is important to consider whether adding unlabeled data improves performance. In the following, I derive a principled objective for semi-supervised learning that decomposes into a sum of two independent losses: a loss defined on the labeled and unlabeled parts of the data, and another, orthogonal loss defined only on the unlabeled parts of the data. Given auxiliary unlabeled data, one can then reduce the variance of the latter loss estimate without affecting the former, hence achieving a variance reduction over using labeled data alone.

Definition 3 (Affine set)**.** *A set Ω is **affine** if for any $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \Omega$ and $a_1, \ldots, a_n \in \mathbb{R}$, we have $\sum_{i=1}^n a_i \mathbf{x}_i \in \Omega$. Every affine set is convex.*

Definition 4 (Generalized Pythagorean Theorem)**.** *(Murata et al., 2004) For $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$ and Ω an affine set, let*

$$P_\Omega(\mathbf{x}) = \arg\min_{\omega \in \Omega} D_F(\omega || \mathbf{x})$$

be the Bregman projection, then

$$D_F(\mathbf{x}_1 || \mathbf{x}_2) = D_F(\mathbf{x}_1 || P_\Omega(\mathbf{x}_2)) + D_F(P_\Omega(\mathbf{x}_2) || \mathbf{x}_2).$$

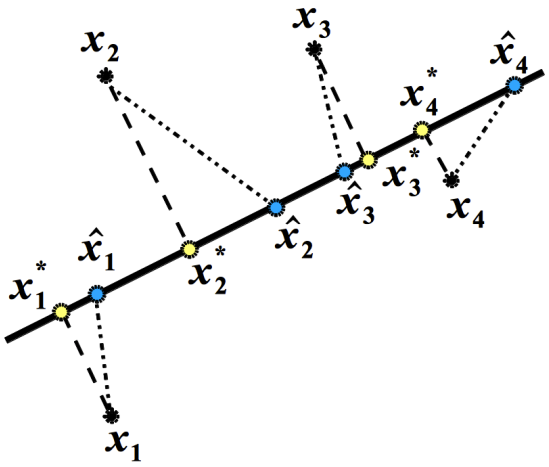


Figure 6.1: Reverse loss decomposition for an identity transfer, making the corresponding Bregman divergence a least squares loss. In this case, for $\hat{\mathbf{x}} = C\mathbf{y}$ the supervised reconstruction of $\mathbf{x}$ using the given label $\mathbf{y}$ and $\mathbf{x}^* = C\phi^*$, satisfies $\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2 = \|\mathbf{x}_i - \mathbf{x}_i^*\|_2 + \|\hat{\mathbf{x}}_i - \mathbf{x}_i^*\|_2$, by the Pythagorean theorem. The generalization of this result beyond the squared norm to general Bregman divergences is given in Theorem 8.

This decomposition for an identity transfer is depicted in Figure 6.1.

For reverse semi-supervised learning, we define the subspace based on the current reverse model, C: $\Omega = \{C\phi \mid C\phi \in \mathbf{f}(\mathcal{X})\}$, where $\mathcal{X}$ is the space of all possible input vectors, x. For an affine set $\mathcal{F}$, $C\mathcal{F}$ is guaranteed to be affine. Examples of transfers that enable $\mathcal{F}$ to be affine include $\mathbf{f}(\mathbf{x}) = x$, $\mathbf{f}(\mathbf{x}) = \log_2(\mathbf{x}) + 1/\log(2)$ (KL divergence) and the relaxed sigmoid, $\mathbf{f}(\mathbf{x}) = \text{sign}(\mathbf{x}) \cdot \log(|x|/\theta + 1)$ ($\theta \in \mathbb{R}$). A sufficient though not necessary condition to make the image affine is for $\mathbf{f}$ to be surjective on $\mathbb{R}^k$. In the following theorem, we provide our variance reduction argument for this class of transfer functions.

Theorem 8. *For any X_l, X_u, Y_l, C and transfer function, $\mathbf{f}$, with resulting affine feature set, $\mathcal{F}$,*

$$E[D_{F^\star}(CY_l \,||\, \mathbf{f}(X_l))]/T_l = E[D_{F^\star}(CY_l \,||\, C\Phi_l^*)]/T_l + E[D_{F^\star}(C\Phi^* \,||\, \mathbf{f}(X))]/T_u \tag{6.2}$$

where $X = [X_l; X_u]$ and $\Phi^* = \arg\min_{Z \in \mathcal{F}} D_{F^\star}(C\Phi||f(X))$.

Proof: From the Generalized Pythagorean Theorem, we know that

$$E[D_{F^\star}(CY_l \,||\, \mathbf{f}(X_l))] = E[D_{F^\star}(CY_l \,||\, C\Phi_l^*)] + E[D_{F^\star}(C\Phi_l^* \,||\, \mathbf{f}(X_l))].$$

Since

$$E[D_{F^\star}(C\Phi_l^*||f(X_l))/T_l] = E[D_{F^\star}(C\Phi^*||f(X))/T_u]$$

we get the desired decomposition. ■

Therefore, with more unlabeled data, the variance of the estimate of the second expected value in the loss decomposition is decreased, reducing the variance of the supervised learning error estimate. With a reduced variance estimate of the loss, one is closer to optimizing the true loss and should

obtain an improved solution. For large amounts of unlabeled data, the simpler semi-supervised learning loss in (6.1) closely approximates (6.2), but introduces a bias due to double counting the unlabeled loss. Directly optimizing (6.2), however, is problematic due to the fact that the parameters are in the second argument of the Bregman divergence. We therefore opt for the close approximation in (6.1) to a theoretically sound objective.

If we only want to consider convex rather than affine sets, the Generalized Pythagoras Theorem changes to an inequality on the losses

$$D_F(\mathbf{x}_1||\mathbf{x}_2) \geq D_F(\mathbf{x}_1||P_\Omega(\mathbf{x}_2)) + D_F(P_\Omega(\mathbf{x}_2)||\mathbf{x}_2).$$

Unfortunately, this inequality does not provide any guarantees for unlabeled data. It may instead be possible to generalize beyond affine sets using the cosine law, which applies to any sets.

Definition 5 (Cosine Law)**.** *For $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathcal{X}$ for any $\mathcal{X} \subset \mathbb{R}^n$, then*

$$D_F(\mathbf{x}_1 \,||\, \mathbf{x}_2) = D_F(\mathbf{x}_1 \,||\, \mathbf{x}_2) + D_F(\mathbf{x}_3 \,||\, \mathbf{x}_2) - (\mathbf{x}_1 - \mathbf{x}_3)'(\nabla F(\mathbf{x}_2) - \nabla F(\mathbf{x}_3))$$

If this extra term $(\mathbf{x}_1 - \mathbf{x}_3)'(\nabla F(\mathbf{x}_2) - \nabla F(\mathbf{x}_3))$ is zero, then we obtain the same variance reduction argument. In general, the cosine law might enable a similar variance reduction argument if specific properties of the convex set and transfer are known that enable the extra term to be bounded.

6.1.2 Algorithms for semi-supervised classification and regression

This formulation for standard semi-supervised learning results in interesting theoretical insights; unfortunately, it is unclear how to take advantage of the convex reformulation techniques for this objective. Instead, we develop classification and regression algorithms by alternating between optimizing Φ and C in Equation (6.1). Though this approach deviates from the focus on global solutions, the algorithms themselves are quite simple and efficient and enables us to explore this regularized factorization objective with interesting theoretical properties.

The semi-supervised regression algorithm is given in Algorithm 12 and the semisupervised classification algorithms in Algorithm 10 and 11. The regression algorithm simply uses an off-the-shelf optimizer (like limited memory BFGS) to alternate between optimizing Φ and C according to the objective provided in Equation 6.1. The classification algorithm, however, requires a more careful treatment due to the constraints. To develop this algorithm, we first start in the unsupervised setting, giving Algorithm 8 and 9, and then extend to semi-supervised learning in Algorithm 10 and Algorithm 11.

For discrete variable, $\Phi \in \{0,1\}^{k \times T}$, Banerjee et al. (2005) show that the optimization for Bregman clustering can be simplified to a simple k-means algorithm. Using a similar insight, we

can formulate a correspondingly efficient form of Bregman divergence clustering *that incorporates a non-linear transfer between $C\Phi$ and Y*:

$$D_{F^\star}(C\Phi||\mathbf{f}(X)) = D_F(X||\mathbf{f}^\star(C\Phi)) = D_F(X||\mathbf{f}^\star(C)\Phi) \tag{6.3}$$

$$\implies \min_{\Phi\in\mathcal{F}} \min_{C\in Dom(\mathbf{f}^\star)} D_{F^\star}(C\Phi||\mathbf{f}(X)) = \min_{\Phi\in\mathcal{F}} \min_{M\in Dom(\mathbf{f})\subset\mathbb{R}^{n\times k}} D_F(X||M\Phi) \tag{6.4}$$

$$= \min_{\Phi\in\mathcal{F}} \min_{M\in Dom(\mathbf{f})} \sum_{j=1}^{T} \sum_{i:\Phi_{ij}=1} D_F(X_{:,j}||M_{:,i}\Phi) \tag{6.5}$$

$$= \min_{\Phi\in\mathcal{F}} \sum_{i=1}^{k} \frac{1}{\Phi_{i,:}\mathbf{1}} \sum_{j:\Phi_{ij}=1} X_{i:}. \tag{6.6}$$

The proof for the simplification of the inner maximization is given in Appendix F.1. Algorithm 8 illustrates the modified Bregman clustering algorithm which now permits non-linear transfers. The derivation for soft constraints, $\Phi \in [0,1]^{k\times T}$, is given in Appendix F.1, with extensions to the semi-supervised setting. The clustering algorithms could similarly incorporate kernels; for brevity, using kernels is only demonstrated for the semi-supervised regression algorithm.

Algorithm 8 Non-linear Bregman Hard Clustering

Input: X, k, D_F
Output: reverse model C and hard clusters Φ

Initialize M (e.g. k randomly selected columns from X)
while (change in $D_F(X||M)) >$ tolerance **do**
  E-Step: $\forall t \in \{1,\ldots,T\}$:
    $\Phi_{jt} = 1 \quad \text{for} \quad j = \arg\min_i D_F(X_{:t}||M_{i:})$
  M-Step: $\forall i \in \{1,\ldots,k\}$:
    $M_{i:} = \frac{1}{\Phi_{i:}\mathbf{1}} \sum_{t:\Phi_{it}=1} X_{:t}$
end while
// W could now also be computed, such as with $\min_W D_{F^\star}(XW \;||\; \mathbf{f}(\Phi))$
return $C = \mathbf{f}(M),\ \Phi$

Algorithm 9 Non-linear Bregman Mixture Models

Input: X, k, D_F and smoothness parameter ρ; as $\rho \to \infty$, becomes a hard clustering algorithm
Output: reverse model C and soft clusters Φ

Initialize M (e.g. k randomly selected columns from X)
$\text{err}(M,p) = -\sum_t \log\left(\sum_i p_i \exp(-\rho D_F(X_{:t}||M_{i:}))\right)$
while (change in $\text{err}(M,P)) >$ tol **do**
  E-Step: $\Phi_{it} = p_i \exp[-\rho(D_F(X_{:t}||M_{i:})]$
    $\Phi_{:t} = \Phi_{:t} / \sum_i \Phi_{it}$ //normalize Φ
  M-Step: $M = \text{diag}(\Phi\mathbf{1})\Phi X'$
    $p = \frac{1}{T}\Phi\mathbf{1}$
end while
return $C = \mathbf{f}(M),\ \Phi$

Algorithm 10 Reverse Semi-supervised Hard Clustering

Input: $X_l, X_u, Y_l, \boldsymbol{\mu}, D_F$, where $Y_l \in \{0,1\}^{k \times T}$
Output: C, Y

$Y = [Y_l \ \ \mathbf{0}]$
$X = [X_l \ X_u]$
Initialize M (e.g. k randomly selected columns of X or with Algorithm 8)
$\text{err}(M, Y) = \sum \frac{1}{T_l} \sum_{t=1}^{T_l} D_F(X_{:t} || M_{(i:Y_{it}=1),:})) + \frac{\boldsymbol{\mu}}{T_u} \sum_{t=T_l+1}^{T_l+T_u} D_F(X_{:t} || M_{(i:Y_{it}=1),:}))$
while (change in $err(M, Y)) >$ tolerance **do**
  E-Step: $\forall t \in \{T_l + 1, \ldots, T_l + T_u\}$
    $Y_{:t} = \mathbf{0}$ // clear current label for sample t
    $Y_{jt} = 1 \quad \text{for} \quad j = \arg\min_i D_F(X_{:t} || M_{i:})$
  M-Step: $\forall i \in \{1, \ldots, k\}$
    $M_{i:} = \frac{1}{Y_{i:}\mathbf{1}} \sum_{t:Y_{it}=1} X_{:t}$
end while
return $C = \mathbf{f}(M), Y$

Algorithm 11 Reverse Semi-supervised Soft Clustering

Input: $X_l, X_u, Y_l \in [0,1]^{k \times T_l}$, forward loss D_F, unsupervised-weighting parameter $\boldsymbol{\mu}$ and smoothness parameter ρ, where as $\rho \to \infty$, this algorithm becomes a hard clustering algorithm
Output: C, Y

$\boldsymbol{\lambda} = [\mathbf{1} \ \ \boldsymbol{\mu}]$ // weighting on samples
Initialize M (e.g. k randomly selected columns of X or with Algorithm 9)
$p = \mathbf{1}/k$
$Y = [Y_l \ \ \mathbf{0}]$
$X = [X_l \ X_u]$
$\text{err}(M, p) = -\sum_t \log\left(\sum_i p_j \exp(-\rho \boldsymbol{\lambda} D_F(X_{:t} || M_{i,:}))\right)$
while (change in $\text{err}(M, p)) >$ tolerance **do**
  E-Step: $\forall t \in \{T_l + 1, \ldots, T_l + T_u\}$
    $\forall i \in \{1, \ldots, k\}:$
      $\Phi_{i,t} = p_i \exp(-\rho\mu(D_F(X_{:t} || M_{i,:}))$
    $\Phi_{:t} = \Phi_{:t} / \sum_i \Phi_{i,t}$ // normalize to ensure sums to 1
  $Y = [Y_l \ \Phi]$
  M-Step: $\forall i \in \{1, \ldots, k\}$
    $M = \text{diag}(Y\mathbf{1}) Y X'$
    $p = \frac{1}{t} Y \mathbf{1}$
end while
return $C = \mathbf{f}(M), Y$

Algorithm 12 Reverse Semi-supervised Regression

Input: $X_l, X_u, Y_l, \boldsymbol{\mu}, D_F, D_{F^\star}, \alpha$, kernel
Output: forward model A, basis B and imputed outputs Y

// Reasonable options for parameters are $\alpha \leq 0.1$ and kernel$(X, X) = X'X$
$\boldsymbol{\lambda} = [\mathbf{1} \;\; \boldsymbol{\mu}]$ // weighting on samples
$X = [X_l \; X_u], \; K = \text{kernel}(X, X)$
Initialize Φ and B, likely randomly or using initial supervised model only on labeled data
$\text{err}(B, \Phi) = \boldsymbol{\lambda} D_{F^\star}(B \; [Y_l \;\; \Phi] \; || \; \mathbf{f}(K))$
while (change in $\text{err}(B, \Phi)) >$ tolerance **do**
    $B = \text{argmin}_B \, \text{err}(B, \Phi)$
    $\Phi = \text{argmin}_\Phi \, \text{err}(B, \Phi)$
end while
$Y = [Y_l \; \Phi]$
// Using the imputed targets, compute the corresponding forward solution
$A = \text{argmin}_A \, D_F(AK \; || \; \mathbf{f}^{-1}(Y)) + \alpha \, \text{tr}(A'AK)$
return A, B, Y

6.1.3 Experimental results

In this section, we explore two main points: (1) the utility of non-linear transfers, and (2) the general performance of our principled semi-supervised algorithm. Synthetic data with specific transfers is used to assess the importance of having the correct transfer function. Results are also reported on real-world data. I report transductive error as some competitors are solely transductive.

Regression Experiments: For regression, synthetic data is generated with three transfer functions: (i) $Y = WX$, (ii) $Y = (WX)^3$ and (iii) $Y = \exp(WX)$. The data is generated in reverse with Y and C generated from $\mathcal{N}(\mathbf{0}, I)$ and X set to $X = \mathbf{f}^{-1}(CY + \text{noise})$. I also report results on three UCI datasets: kin-32fh (n = 34, k = 1), puma-8nm (n = 8, k =1) and California housing (n = 5, k = 1). I compare again transductive regression (Cortes and Mohri, 2007) and supervised (kernel) least-squares as a baseline comparison. Limited memory BFGS is used to optimize the objectives.

Parameters are tuned using transductive error on a portion of the unlabeled data for each algorithm on each dataset. In practice, parameters are usually tuned using cross-validation on only labeled examples; this approach, however, can have confounding effects due to the lack of labeled data for evaluation. Using only a subset of the unlabeled data reduces overfitting, while enabling more accurate parameter selection for each algorithm. I tuned the trade-off parameter $\mu \in \{\text{1e-3}, \; \text{1e-2}, \; \text{1e-1}\}$ (above $\mu = 1$, performance degrades). For transductive regression, we tuned over $\lambda, \; C_1$ and C_2 and fixed r as recommended in their chapter. All algorithms were tuned over using no kernel, a linear kernel and Gaussian kernels with widths in $\{0.01, \; 0.1, \; 1, \; 5, \; 10\}$.

The results in Tables 6.1 and 6.2 clearly indicate that using the correct transfer function is crucial to performance. For each of the synthetic datasets, optimizing with the transfer used to generate

	SYN-GAUSS N=30, K=5, $T_u = 200$	SYN-CUBED N=20, K=3, $T_u = 200$	SYN-EXP N=5, K=2, $T_u = 200$
SUP-KERNEL	**2e-14 ± 4.4e-15**	0.246 ± 0.030	408 ± 116.9
TRANS-REG	3E-06 ± 8.9E-07	0.244 ± 0.030	1039 ± 136.2
SEMI EUC	**6e-31 ± 8.9e-32**	0.120 ± 0.003	1423 ± 95.61
SEMI CUBED	1.91 ± 0.752	**3e-5 ± 2.6e-06**	348.1 ± 34.03
SEMI EXP	UNSTABLE	UNSTABLE	**1.7e-4 ± 0.000**

Table 6.1: Average transductive error of semi-supervised regression techniques on synthetic dataset, with (n, k, t_u) and $t_l = 20$, over 50 splits of the data.

	KIN-32FH N=34, K=1, T_u =100, T_l = 20	PARKINSONS N=9, K=1, T_u =200, T_l = 10	CALHOUSING N=5, K=1, T_u =300, T_l=50
SUP-KERNEL	0.305± 0.010	89.89 ± 2.951	134.2 ± 1.721
TRANS-REG	0.327 ± 0.009	98.19 ± 3.630	**119.6 ± 4.758**
LAP-RLS	0.199 ± 0.005	**75.02 ± 0.9897**	**127.3 ± 1.486**
SEMI EUC	0.269 ± 0.006	79.03 ± 1.391	134.7 ± 1.635
SEMI CUBED	**0.185 ± 0.004**	213.8 ± 67.05	131.8 ± 1.721
SEMI EXP	**0.186 ± 0.004**	UNSTABLE	147.5 ± 2.269

Table 6.2: Average transductive error of semi-supervised regression techniques on a variety of real datasets, over 50 splits of the data.

the synthetic data performs statistically significantly better than the other algorithms and objectives, verifying our expectations. The synthetic results for the exponential transfer are particularly illustrative: errors are amplified for non-exponential data, but are much lower for exponential data.

These insights and properties transfer to performance on real datasets. On the kin-32fh dataset, using an exponential transfer considerably improved performance. This surprising improvement likely occurred because the kin-32fh outputs were all positive and the exponential transfer enforces $Y_{ij} \in \mathbb{R}^+$. On the mainly linear simulated robot-arm dataset, puma-8nm, the three linear approaches are comparable. The highly nonlinear California housing dataset, however, illustrates some interesting phenomena. First, the addition of kernels is important for modeling: the transductive regression algorithm leverages the wide range of kernels well for modeling (as reducing the number of widths causes it's performance to degrade below supervised learning). Second, the nonlinear cube transfer performed the best out of the three transfers, suggesting that having a range of transfers can positively impact performance.

Though omitted for brevity, running the algorithms using only supervised data resulted in significantly lower performance, particularly for the non-identity transfers. For example, for kin-32fh, the solely supervised solution with an exponential transfer obtained an error of 0.358 ± 0.016 and

the cubed transfer an error of 0.299 ± 0.010 and a less pronounced difference of 0.304 ± 0.010 for the identity transfer.

	SYN-GAUSS N=30, K = 5, T_u =100	SYN-SIGMOID N=10, K=3, T_u =100	SYN-SOFTMAX N=10, K = 3, T_u =100
SUP-KERNEL	0.188 ± 1.73E-3	0.290 ± 1.99E-4	0.440 ± 2E-4
LGC	0.445 ± 2.06E-3	0.520 ± 4.98E-5	0.541 ± 5E-4
LAPSVM	0.072 ± 1.56E-3	**0.045 ± 1.49e-4**	0.540 ± 1E-4
LAPRLSC	0.063 ± 9.95E-5	**0.050 ± 3.36e-10**	0.549 ± 0.001
HARD-CLUSTER EUCLIDEAN	**0.027 ± 2.99e-4**	0.056 ± 5.47E-4	0.413 ± 0.002
HARD-CLUSTER SIGMOID	0.144 ± 9.62E-4	0.056 ± 5.97E-4	0.479 ± 0.001
HARD-CLUSTER SOFTMAX	0.165 ± 1.46E-3	0.055 ± 2.49E-4	0.391 ± 0.003
HARD-CLUSTER SIGMOID NC	0.067 ± 5.64E-4	0.055 ± 2.49E-4	0.430 ± 0.004
SOFT-CLUSTER EUCLIDEAN	**0.034 ± 9.62e-4**	**0.049 ± 5.54e-4**	0.414 ± 0.001
SOFT-CLUSTER SIGMOID	0.057 ± 5.98E-4	**0.043 ± 3.87e-4**	0.530 ± 1E-4
SOFT-CLUSTER SOFTMAX	0.428 ± 6.91E-3	**0.045 ± 2.81e-4**	**0.358 ± 0.005**
SOFT-CLUSTER SIGMOID NC	0.056 ± 7.77E-4	**0.044 ± 3.94e-4**	0.448 ± 0.001

Table 6.3: Average transductive percent misclassification error of semi-supervised classification techniques on synthetic data, given (n, k, t_u) and $t_l = 10$, over 20 splits. Euclidean, Sigmoid and Softmax correspond to objectives with identity, sigmoid and softmax transfers. Hard/Soft Cluster Sigmoid NC is Bregman normalized cut with a sigmoid transfer.

	YEAST N=8, K = 10, T_u =200, $t_l = 10$	LINK N=1051, $T_u = 200$, $t_l = 10$	SETSTR N=15, T_u =400, $t_l = 50$
SUP-KERNEL	0.528 ± 0.007	0.207 ± 0.029	0.501 ± 0.007
LGC	0.527 ± 0.006	0.150 ± 0.008	0.476 ± 0.014
LAPSVM	0.643 ± 0.005	0.136 ± 0.011	0.479 ± 0.009
LAPRLSC	0.515 ± 0.007	0.144 ± 0.016	0.504 ± 0.012
HARD-CLUSTER EUCLID	**0.487 ± 0.005**	0.130 ± 0.007	0.481 ± 0.019
HARD-CLUSTER SIGMOID	0.634 ± 0.006	0.130 ± 0.007	**0.445 ± 0.015**
HARD-CLUSTER SFTMAX	0.609 ± 0.004	0.134 ± 0.007	0.503 ± 0.017
HARD-CLUSTER SIG NC	0.581 ± 0.006	0.130 ± 0.007	0.510 ± 0.018
SOFT-CLUSTER EUCLID	0.564 ± 0.007	**0.105 ± 0.007**	0.489 ± 0.013
SOFT-CLUSTER SIGMOID	0.606 ± 0.012	0.193 ± 0.030	0.481 ± 0.019
SOFT-CLUSTER SFTMAX	0.555 ± 0.007	0.231 ± 0.027	0.481 ± 0.019
SOFT-CLUSTER SIG NC	0.502 ± 0.007	**0.114 ± 0.006**	0.480 ± 0.014

Table 6.4: Average transductive percent misclassification error of semi-supervised classification techniques on real-world datasets over 50 splits. Euclidean, Sigmoid and Softmax correspond to objectives with identity, sigmoid and softmax transfers. Hard/Soft Cluster Sigmoid NC is Bregman normalized cut with a sigmoid transfer. LINK and SetStr have $k = 2$.

Classification Experiments: Synthetic data was generated with three transfer functions:

(i) $Y = WX$, (ii) $Y = (1 + \exp(-WX))^{-1}$ (sigmoid) and (iii) $Y = \exp(WX)(\mathbf{1}^T \exp(WX))^{-1}$

(softmax). The data was generated by selecting uniformly random classes for Y and drawing C from a Gaussian distribution for the identity transfer and from a uniform distribution for sigmoid and softmax, with rows of C and the noise normalized for softmax. For sigmoid and softmax, $X = \mathbf{f}^{-1}((1-\sigma)CY + \sigma \cdot \text{noise})$. I also tested on three real datasets: the Yeast dataset with (n=8, k=10), LINK, a WebKB dataset with (n=1051, k=2), and SetStr, a semi-supervised benchmark dataset with (n = 15, k=2).

I compare to three semi-supervised algorithms: learning with local and global consistency (LGC) (Zhou et al., 2004), Laplacian SVMs (Sindhwani et al., 2005) and Laplacian regularized least squares (RLSC) (Sindhwani et al., 2005). The Bregman hard and soft-clustering objectives are tested for a variety of transfer functions, kernels and instance weights. I tuned the trade-off parameter, $\mu \in \{\text{1e-3},\ \text{1e-2},\ \text{1e-1}\}$ and soft-clustering parameter, $\rho \in \{1,\ 10,\ 50,\ 100,\ 200\}$. For LGC, I tuned the smoothness parameter $\alpha \in \{\text{1e-5},\ \text{1e-3},\ 0.1,\ 0.5\}$. For the Laplacian algorithms, I set $\gamma_A = \text{1e-6}$ and $\gamma_I = 0.01$ and tuned the Laplacian degree in $\{1,\ 2,\ 5\}$ and number of nearest neighbours in $\{5,\ 20\}$. All algorithms were tuned over no kernel, a linear kernel and a Gaussian kernel with widths in $\{0.01,\ 0.1,\ 1,\ 5,\ 10\}$.

As with regression, the transfer has an impact on performance (Tables 6.3 and 6.4). Though all algorithms performed well on the Gaussian data, we can see that the incorrect sigmoidal and softmax transfers did decrease performance. We see a corresponding result for the synthetic sigmoidal and softmax data. Interestingly, for the sigmoidal transfer on the Yeast and LINK datasets, the normalized cut extension improved performance. Moreover, it seems to be the case that the normalized cut extension performs better with the soft-clustering algorithm rather than the hard-clustering algorithm. On the three real-world datasets, it is interesting that there is no obvious trend of hard over soft-clustering or one transfer over another. On Yeast, hard-clustering with an identity transfer (Euclidean loss) performs the best; on LINK, soft-clustering with a euclidean transfer performs the best; and on SetStr, hard-clustering with a sigmoidal transfer performs the best. Overall, we can see that the variety of options available under regularized factor models enables us to tailor our objective to the given data. Using prior knowledge alongside empirical selection of transfers, prediction accuracy can be dramatically improved.

6.2 Convex representation-imputed semi-supervised learning

In this second semi-supervised learning formulation, the unlabeled data is used to learn a "better" representation; that representation can then be used to improve learning on the small, labeled dataset. A more useful representation would constitute one that improves prediction accuracy, improve generalization and/or speeds learning on the labeled data. Properties of representations that intuitively

promote these three goals are lower-rank features, sparse features, and local features; as such, these properties are widely sought in representation learning. As discussed in Chapter 5, however, there is a lack of theoretical development on the precise meaning of a "better" representation. It is more difficult, therefore, to gauge the effect of the unlabeled data on supervised learning performance and we can no longer guarantee that unlabeled data does not actually harm performance. In practice, however, despite the lacking theoretical support, representation-imputed semi-supervised learning has been shown to provide significant increases in prediction performance (Pereira and Gordon, 2006; Rish et al., 2008; Lee et al., 2009).

As with standard semi-supervised learning, solving representation-imputed semi-supervised learning optimizations has been an important computational challenge. Many previous formulations use local training methods, including staged training procedures that separate the unsupervised from the supervised phase (Lee et al., 2009) and alternating iterative algorithms for semi-supervised dimensionality reduction (Rish et al., 2008; Pereira and Gordon, 2006). More recently, a convex formulation (Goldberg et al., 2010) was developed for transductive dimensionality reduction; though an important advance, this formulation 1) is limited to the transductive setting and so does not enable extraction of the representation Φ nor the prediction model and 2) cannot enforce individual constraints on the supervised and unsupervised parts of the model. In the following sections, I present the first convex formulation for representation-imputed semi-supervised learning that enables both of these properties, and conclude with an empirical demonstration.

6.2.1 Convex reformulation

Consider a setting where we are given an $n \times T_u$ matrix of unlabeled data X_u, an $n \times T_l$ matrix of labeled data X_l, and a $r \times T_l$ matrix of target values Y_l. We would like to learn a $k \times (T_l + T_u)$ representation matrix $\Phi = [\Phi_l, \Phi_u]$ and an $n \times k$ basis dictionary $C^{(1)}$ such that $X = [X_l, X_u]$ can be reconstructed from $\hat{X} = C^{(1)}\Phi$, while simultaneously learning an $r \times k$ prediction model $C^{(2)}$ such that Y_l can be reconstructed from $\hat{Y}_l = C^{(2)}\Phi_l$. Let $L_u\left(C^{(1)}\Phi\ ;\ X\right)$ and $L_s\left(C^{(2)}\Phi_l\ ;\ Y_l\right)$ denote unsupervised and supervised losses respectively, which we assume are convex in their first argument. To avoid degeneracy we impose the constraints ${C^{(1)}}_{:j} \in \mathcal{C}^{(1)}$ and ${C^{(2)}}_{:j} \in \mathcal{C}^{(2)}$ for bounded closed sets $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$.

To formulate this joint training problem convexly, we can notice that this problem is similar to the two-view learning problem from Section 5.3, with input data as one view and targets as the other. In this setting, however, the corresponding instances of the second view are not available for the unlabeled data. The loss, therefore, has to be carefully expressed to ensure that (1) the labeled views are matched and (2) the targets for the unlabeled data can be inputted. The general semi-

supervised learning problem is expressed in the next proposition; specific choices of regularizers are discussed after this general result.

Proposition 9 (Convex representation-imputed semi-supervised learning)**.** *For bounded closed sets $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$ such that span$(\mathcal{C}^{(1)}) = \mathbb{R}^n$ and span$(\mathcal{C}^{(2)}) = \mathbb{R}^r$, convex loss L, and assuming desired representation properties are specified by the norm $\|\cdot\|_{\diamond,1}$, representation-imputed semi-supervised learning can be formulated as a convex optimization:*

$$\min_{C^{(1)}\in\mathcal{C}^{(1)\infty}} \min_{C^{(2)}\in\mathcal{C}^{(2)\infty}} \min_{\Phi_l,\Phi_u} L_u\left(C^{(1)}[\Phi_l,\Phi_u]\ ;\ X\right) + L_s\left(C^{(2)}\Phi_l\ ;\ Y_l\right) + \alpha\|[\Phi_l,\Phi_u]\|_{\diamond,1} \tag{6.7}$$

$$= \min_{C\in\mathcal{C}^{\infty}} \min_{\Phi} L\left(C\Phi\ ;\ \begin{bmatrix} X_l & X_u \\ Y_l & 0 \end{bmatrix}\right) + \alpha\|\Phi\|_{\diamond,1} \tag{6.8}$$

$$= \min_{Z} L\left(Z\ ;\ \begin{bmatrix} X_l & X_u \\ Y_l & 0 \end{bmatrix}\right) + \alpha\|Z'\|^*_{(\mathcal{C},\diamond^*)}, \tag{6.9}$$

where $C = \begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix}$, $Z = C[\Phi_l,\Phi_u]$, $\mathcal{C} = \mathcal{C}^{(1)} \times \mathcal{C}^{(2)}$ *and*
$L\left(Z\ ;\ \begin{bmatrix} X_l & X_u \\ Y_l & 0 \end{bmatrix}\right) = L_u\left(Z_{(1:n),:}\ ;\ [X_l\ X_u]\right) + L_s\left(Z_{(n+1:2n),(1:T_l)}\ ;\ Y_l\right)$.

Proof: The proof of this proposition follows immediately from Theorem 2, since if $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$ are bounded closed sets that span $\mathbb{R}^n$ and $\mathbb{R}^r$, then $\mathcal{C} = \mathcal{C}^{(1)} \times \mathcal{C}^{(2)}$ is also closed and bounded with span$(\mathcal{C}) = \mathbb{R}^{n+r}$. ■

The optimization in (6.7) is a regularized factor model, where the data is now a block matrix including both the input data and target data as shown in (6.8). The resulting Z from optimizing (6.8) will be a transductive matrix completion solution, where the part of Z corresponding to the missing labels Y_u (set to zero in the loss for the missing entries) will correspond to the predicted targets for the unlabeled data. Note that the combined loss L ignores the difference between $\mathbf{0}$ and the corresponding component of Z; otherwise, it would skew learning to push Z to equal zero for those entries. This loss that ignores a component of Z is still convex, with a zero gradient value for the components of Z corresponding to the zero entries.

Unlike previous staged training procedures (Lee et al., 2009) and alternating minimizations (Rish et al., 2008; Pereira and Gordon, 2006) that separate the unsupervised from the supervised phase, Proposition 9 provides a jointly convex formulation that allows all components to be trained simultaneously. As before, however, not every choice of $\mathcal{C}$ and $\|\Phi\|_{\diamond,1}$ will result in an efficiently computable optimization and recovery procedure. From Section 4.3, we know that there are two settings where the formulation is efficient for a partitioned constraint on C: 1) a (1,1)-block norm, $\|\Phi\|_{1,1}$, imposing sparse structure on Φ and 2) a (2,1)-block norm, $\|\Phi\|_{2,1}$, pushing entire rows of Φ to zero to obtain a lower-dimensional representation.

Sparse Coding Semi-supervised Formulation: From Theorem 6, we know that for

$$\|\Phi\|_{1,1},\ \mathcal{C}^{(1)} = \{\mathbf{c}^{(1)}\ : \|\mathbf{c}^{(1)}\|_{q_1} \leq \beta_1\},\ \mathcal{C}^{(2)} = \{\mathbf{c}^{(2)}\ : \|\mathbf{c}^{(2)}\|_{q_2} \leq \beta_2\}$$

we obtain the efficiently computable induced norm

$$\|Z'\|^*_{(\mathcal{C}^{(1)} \times \mathcal{C}^{(1)}, \infty)} = \sum_t \max\left(\frac{1}{\beta_1}\|Z^{(1)}{}_{:t}\|_{q_1}, \frac{1}{\beta_2}\|Z^{(2)}{}_{:t}\|_{q_2}\right).$$

Moreover, from Proposition 8, we know that C and Φ can be recovered from the optimal Z as $\Phi_{t,t} = \max(\|Z^{(1)}{}_{:t}\|_{q_1}, \frac{1}{\beta_2}\|Z^{(2)}{}_{:t}\|_{q_2})$ and $C = Z\Phi^{-1}$. Unfortunately, as in the unsupervised case, we reach the conclusion that $\|\Phi\|_{1,1}$ regularization leads to a trivial form of vector quantization, and does not impose interesting structure on the representation.

Subspace Learning Semi-supervised Formulation: Fortunately, the situation for subspace learning is more interesting. If instead we choose

$$\|\Phi\|_{2,1},\ \mathcal{C}^{(1)} = \{\mathbf{c}^{(1)} \in \mathbb{R}^n\ :\ \|\mathbf{c}^{(1)}\|_2 \leq \beta_1\},\ \mathcal{C}^{(2)} = \{\mathbf{c}^{(2)} \in \mathbb{R}^n\ :\ \|\mathbf{c}^{(2)}\|_2 \leq \beta_2\}$$

we get induced norm

$$\|Z'\|^*_{(\mathcal{C}^{(1)} \times \mathcal{C}^{(2)}, 2)} = \max_{0 \leq \eta \leq 1} \|E_\eta^{-1} Z\|_{tr}$$

where

$$E_\eta := \begin{bmatrix} \beta_1/\sqrt{\eta}\, I_n & 0 \\ 0 & \beta_2/\sqrt{1-\eta}\, I_n \end{bmatrix}.$$

This setting provides a novel and effective convex formulation of representation-imputed semi-supervised learning with dimensionality reduction. There are several ways to optimize this objective. To date, the most efficient way is to use the procedure described for multi-view subspace learning, which solves the same optimization. Using the semisupervised target, $\begin{bmatrix} X_l & X_u \\ Y_l & \mathbf{0} \end{bmatrix}$, in the objective in Algorithm 7, instead of $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$, therefore, gives an efficient solution method for convex subspace semi-supervised learning. In the following experiments, before this more efficient solution approach was known, we originally solved the dual of the problem, and then recovered optimal Z to the primal problem (Zhang et al., 2011); the resulting solutions, however, are equivalent.

6.2.2 Experimental results

Algorithms: To evaluate the proposed convex representation-imputed semi-supervised learning method (which I call S-RFM for semi-supervised regularized factor models), we compared its performance to two local and one convex approach respectively: alternation (ALT), staged-alternation

(STAGE) and a transductive matrix completion method (Goldberg et al., 2010). In the alternating approach, two of the three variables, $C^{(1)}, C^{(2)}, \Phi$, are fixed and the other optimized, repeating optimization over each variable until convergence. In the staged-alternator, the optimizer alternates between $C^{(1)}$ and Φ until convergence and then optimizes the prediction model, $C^{(2)}$. Note that the staged-alternator is the approach taken by Lee et al. (2009); however, we included their implementation in the results for completeness.[1] The implementation of the transductive matrix completion method follows the settings outlined by Goldberg et al. (2010). Though this method is convex, the formulation does not provide extraction of the representation Φ nor the prediction model $C^{(2)}$—instead it only recovers the analog of Z containing transductive predictions on the unlabeled data. Moreover, it cannot enforce individual constraints on the unsupervised and supervised parts of the model, $C^{(1)}$ and $C^{(2)}$ respectively.

Datasets: We investigated six classification datasets: (i) A synthetic dataset with features and labels generated analogously to Goldberg et al. (2010), which contains 20 features and 400 samples. The rank of the generated feature matrix is 4, and zero mean independent Gaussian noise with variance $\sigma^2 = 0.1$ was added to the features. (ii) A UCI dataset, Wisconsin Breast Cancer (WBC), which contains 10 features and 683 samples.[2] (iii) A UCI dataset, Ionosphere, which contains 34 features and 351 samples.[3] (iv) Three semi-supervised learning benchmark datasets, BCI, COIL and g241n, which collectively contain 1500 samples, with 117, 241 and 241 features respectively.[4] For experiments on transductive learning, we randomly selected from each dataset T_l examples as the labeled data and T_u examples as the unlabeled data. Both T_l and T_u are reported in Table 6.6. Then we measured the transductive classification error on the T_u examples, and this error was further averaged over runs on five different random choices of the T_l and T_u examples.

Parameter selection: Because the algorithms are similar, with the same parameters, cross validation on such a small number of labeled samples results in artificially large differences. To mitigate this problem, therefore, we focus the comparison on the best possible performance of each algorithm across a range of parameter settings.

Comparison 1: Optimization We first investigated the differences in objective values, by setting the loss functions and parameters to common choices across the algorithms. In particular, we set the supervised loss to the least-squares loss, the unsupervised loss to the logistic loss, and the regularizer

[1] http://www.eecs.umich.edu/∼honglak/softwares/fast_sc.tgz
[2] http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)
[3] http://archive.ics.uci.edu/ml/datasets/Ionosphere
[4] http://www.kyb.tuebingen.mpg.de/ssl-book/benchmarks.html

Method	$\mu = 10$		$\mu = 0.1$	
	$\alpha = 0.02$	runtime (sec)	$\alpha = 0.02$	runtime (sec)
COIL				
S-RFM	0.70	156	0.6809	93
ALT	0.72	2084	0.6951	3439
STAGE	30.052	94	0.6934	597
WBC				
S-RFM	1.13	27	0.6711	4
ALT	1.14	292	0.6796	872
STAGE	1.19	166	06981	163
BCI				
S-RFM	0.69	43	0.6483	12
ALT	0.69	288	0.6534	528
STAGE	0.74	105	0.6936	158
IONOSPHERE				
S-RFM	0.78	75	0.6434	5
ALT	0.78	167	0.6477	320
STAGE	0.84	71	0.6946	103
G241N				
S-RFM	0.70	111	0.6809	94
ALT	2.17	1886	0.6966	3782
STAGE	30.54	81	0.6934	568
SYNTHETIC				
S-RFM	0.90	15	0.6503	4
ALT	0.90	155	0.6556	384
STAGE	0.94	83	0.6956	128

Table 6.5: Minimum objective values in Equation (6.7) obtained by the training methods on six data sets. The objective values is always the lowest for S-RFM, though sometimes the alternator and the staged algorithms achieve this global minimum. When ALT or STAGE obtain the minimum error, however, their runtime is always worse than the runtime for S-RFM to obtain the same solution.

	COIL (n=241, T_l=10, T_u=100)	WBC (n=10, T_l=10, T_u=50)	BCI (n=117, T_l=10, T_u=200)
ALT	0.464 ± 0.036	0.388 ± 0.156	0.440 ± 0.028
STAGED	0.476 ± 0.037	0.200 ± 0.043	0.452 ± 0.041
LEE	0.414 ± 0.029	0.168 ± 0.100	0.436 ± 0.093
GOLDBERG	0.484 ± 0.068	0.288 ± 0.105	0.540 ± 0.025
S-RFM	0.388 ± 0.043	0.134 ± 0.072	0.380 ± 0.069

	IONOSPHERE (n=34, T_l=10, T_u=300)	G241N (n=241, T_l=10, T_u=100)	SYNTHETIC (n=20, T_l=40, T_u=360)
ALT	0.457 ± 0.075	0.478 ± 0.053	0.464 ± 0.019
STAGED	0.335 ± 0.050	0.484 ± 0.050	0.417 ± 0.035
LEE	0.350 ± 0.042	0.452 ± 0.073	0.411 ± 0.027
GOLDBERG	0.338 ± 0.053	0.524 ± 0.022	0.496 ± 0.018
S-RFM	0.243 ± 0.042	0.380 ± 0.036	0.341 ± 0.013

Table 6.6: Average test (transductive) classification error of semi-supervised techniques on a variety of real datasets and one synthetic dataset ($\pm$ standard deviation).

to $\|\Phi\|_{2,1}$. S-RFM used L-BFGS to optimize the objective. For the local optimization methods ALT and STAGE a projected gradient method was used to enforce norm constraints on $C^{(1)}$ and a constrained optimization was used for the infinity norm on $C^{(2)}$. To evaluate optimization quality, we fixed $\beta_1 = \beta_2 = 1$, $\alpha = 0.02$ and modified the weight $\mu \in \{0.1, 10\}$ on the unlabeled data loss, $L_u(\cdot\,;\,X)$. Two thirds of the examples were used as labeled data while the rest used as unlabeled data. Table 6.5 shows that S-RFM outperforms the non-convex approaches in terms of both the objective value attained and the training cost on all the data sets. Interestingly, ALT and STAGE occasionally find a solution that is very close to the global optimum.

Comparison 2: Transductive error We evaluated the transductive generalization error for the classification datasets attained by the different methods. In this case, we considered different choices for the loss functions, and report the results for the best, using either a soft-margin support vector machine (hinge loss) or smooth logistic loss for the prediction model; and either projecting or not projecting $C^{(1)}$ and $C^{(2)}$ in the local optimization methods. Limited memory BFGS was used in all cases with a smooth loss function, excluding LEE which uses a conjugate gradient method with a smooth ϵ-L_1 regularizer. In Table 6.6, the best results for the alternators were obtained with a hinge loss with an unprojected optimization.

One can see that in every case S-RFM is either comparable to or outperforms the other competitors. Surprisingly, though the approach developed by Goldberg et al. (2010) is the most similar to S-RFM, it performs noticeably worse than S-RFM. ALT performs poorly for WBC and Ionosphere; one possible reason is that the mixed supervised classification and unsupervised regression

losses create poor local minima. This suggests that for an alternating minimization approach, separating the problem into a factorization step and a classification learning step is more appropriate. We can also see that LEE often performs better than STAGE, despite having the same objective; this result is likely due to optimizations in their code, such as the smoothed sparse regularizer.

6.3 Summary

This chapter explored the impacts of representing two types of semi-supervised learning as regularized factor models. For the first type, standard semi-supervised learning, representing both the supervised and unsupervised components as regularized factorizations (i.e., with reverse prediction) enabled the development of a principled algorithm with variance reduction guarantees. For the second type, representation-imputed semi-supervised learning, representing the problem as a two-view regularized factorization enabled a convex reformulation.

In addition to the contributions to semi-supervised learning, this chapter illustrated the usefulness of the myriad of options under Bregman divergences. The experimental results for standard semi-supervised learning illustrated that kernels, instance weighting, and different transfers all impacted performance, without knowing *a priori* which choices would give the best performance. Given the possible combinations, the standard semi-supervised learning algorithm easily outperformed competitors. Nevertheless, these diverse choices do not yet appear to be widely exploited; improving the understanding and use of these modeling options could significantly impact performance of many machine learning algorithms.

Chapter 7

Autoregressive moving average models using regularized factor models

A central problem in applied data analysis is time series modeling—estimating and forecasting a discrete-time stochastic process—for which the autoregressive moving average (ARMA) and stochastic ARMA (Thiesson et al., 2012) are a fundamental model. An ARMA model describes the behavior of a linear dynamical system under latent Gaussian perturbations (Brockwell and Davis, 2002; Lütkepohl, 2007), which affords intuitive modeling capability, efficient forecasting algorithms, and a close relationship to linear Gaussian state-space models (Katayama, 2006, pp.5-6).

Unfortunately, estimating the parameters of an ARMA model from an observed sequence is a computationally difficult problem: no efficient algorithm is known for computing the parameters that maximize the marginal likelihood of the observed data in an ARMA, stochastic ARMA or linear Gaussian state-space model. Consequently, heuristic local estimators are currently deployed in practice (Hannan and Kavalieris, 1984; Durbin, 1960; Bauer, 2005; Lütkepohl, 2007; Thiesson et al., 2012), none of which provide a guarantee of how well the globally optimal parameters are approximated. For estimating linear Gaussian state-space models, it has been observed that local maximization of marginal likelihood tends to find local optima that yield poor results (Katayama, 2006, Sec. 1.3).

In response to the difficulty of maximizing marginal likelihood, there has been growing interest in *method of moments* based estimators for state-space models, which offer both computationally efficient estimation strategies and sound consistency properties (Andersson, 2009; Hsu et al., 2012; Anandkumar et al., 2012). For ARMA models, the most applicable such estimators are the subspace identification methods for estimating state-space models (Katayama, 2006; Moonen and Ramos, 1993; Van Overschee and De Moor, 1994; Viberg, 1995; Song et al., 2010; Boots and Gordon,

2012). Unfortunately, the statistical efficiency of moment matching generally does not match that of maximum likelihood, which is known to be asymptotically efficient under general conditions (Cramér, 1946, Chapter 33). In fact, there is evidence suggesting that the statistical efficiency of current moment matching estimators is quite weak (Foster et al., 2012; Zhao and Poupart, 2014).

In this chapter, I develop a tractable approach to maximum likelihood parameter estimation for stochastic multivariate ARMA models. To efficiently compute a globally optimal estimate, we re-expressed the problem as a regularized loss minimization and then exploit the convex reformulation approaches developed in Chapter 4. Although there has been recent progress in global estimation for ARMA, such approaches have either been restricted to single-input single-output systems (Shah et al., 2012), estimating covariance matrices for scalar ARMA (Wiesel et al., 2013) or using AR to approximate a scalar ARMA model (Anava et al., 2013). By contrast, the approach developed in this chapter offers the first efficient maximum likelihood based approach to estimating the parameters of a stochastic multivariate ARMA(p, q) model. This convex optimization formulation is general, enabling generalized distributional assumptions and estimation on multivariate data, which has been much less explored than scalar ARMA. An experimental evaluation demonstrates that the globally optimal parameters under the proposed criterion yield superior forecasting performance to alternative estimates, including local minimization of the same criterion, local heuristics for ARMA estimation, and moment-based estimation methods for state-space models.

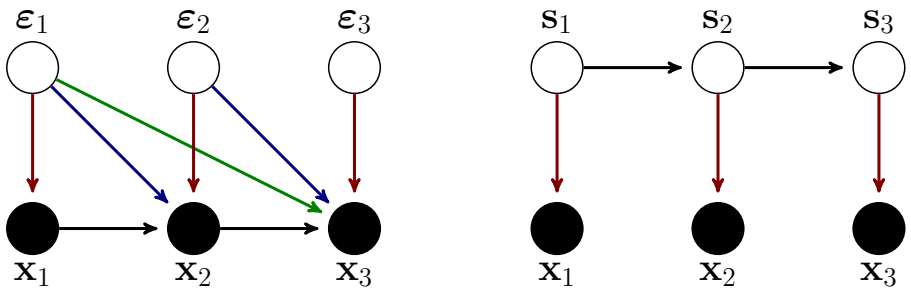
Contributions:

Contribution 1. The first global maximum-likelihood-based approach for estimating autoregressive moving average models, using a tight relaxation on the original problem (White et al., 2015). The idea is to relax the hard constraint requiring the Φ to exactly equal the residual errors: the stochastic ARMA setting. Then, by characterizing the structure of an ARMA model, including the Gaussian assumption on the innovations, as a regularized objective, we can use the advances developed in previous chapters to obtain a convex reformulation, which we call regularized ARMA (RARMA).

Contribution 2. An empirical evaluation that compares to three method-of-moments approaches and three maximum likelihood approaches, on synthetic and real data that indicates that the simple RARMA estimation approach is efficient and outperforms the state-of-the-art in forecasting.

7.1 Background

An ARMA model is a simple generative model of the form depicted in Figure 7.1(a), where the innovation variables, $\boldsymbol{\varepsilon}_t \in \mathbb{R}^k$, are assumed to be i.i.d. Gaussian, $\mathcal{N}(0, \Sigma)$, and the observable



(a) (b)

Figure 7.1: Graphical models depicting the dependence structure of two widely-used temporal models. **(a)** An ARMA$(1, 2)$ model, where the straight down (red) arrows correspond to parameter $B^{(0)}$, the two angled (blue) arrows are $B^{(1)}$ and the longest (green) arrow is $B^{(3)}$. These arrows repeat for $\mathbf{x}_4, \mathbf{x}_5, \ldots$. **(b)** A latent state-space model. These models are equivalent if the state-space model is in observability canonical form (Benveniste et al., 2012, Sec. 6.2.1). Distinct methods are used for estimation in each case depending on whether the variables are discrete or continuous.

variables, $\mathbf{x}_t \in \mathbb{R}^n$, are assumed to be generated by the linear relationship

$$\mathbf{x}_t = \sum_{i=1}^{p} A^{(i)} \mathbf{x}_{t-i} + \sum_{j=1}^{q} B^{(j)} \boldsymbol{\varepsilon}_{t-j} + \boldsymbol{\varepsilon}_t. \tag{7.1}$$

An ARMA(p, q) model is thus parameterized by $A^{(1)}, ..., A^{(p)} \in \mathbb{R}^{n \times n}$, $B^{(1)}, ..., B^{(q)} \in \mathbb{R}^{n \times k}$, and a positive semi-definite matrix Σ; which we simply collect as $\boldsymbol{\theta} = (\{A^{(i)}\}, \{B^{(i)}\}, \Sigma)$.[1]

One classical motivation for ARMA models arises from the Wold representation theorem (Wold, 1938), which states that any stationary process can be represented as an infinite sum of innovations plus a deterministic process that is a projection of a current observation onto past observations: $\mathbf{x}_t = p(\mathbf{x}_t | \mathbf{x}_{t-1}, \ldots) + \sum_{j=0}^{\infty} B^{(j)} \boldsymbol{\varepsilon}_{t-j}$. Thus the autoregressive component of an ARMA model is often motivated as a more parsimonious representation of this Wold representation (Scargle, 1981).

Time series models are used primarily for forecasting: Given an ARMA model with parameters $\boldsymbol{\theta}$, the value of a future observation $\mathbf{x}_{T+h}$ can be predicted from an observed history $\mathbf{x}_{1:T}$ by evaluating $E[\mathbf{x}_{T+h} | \mathbf{x}_{1:T}, \boldsymbol{\theta}]$. The key advantage of ARMA is that such forecasts can be computed efficiently; see Appendix G.4.3 for additional details.

Although forecasting is efficient, the problem of estimating the parameters of an ARMA model raises significant computational challenges, which provides the main focus of this chapter. To begin, consider the marginal log-likelihood of an observed history $\mathbf{x}_{1:T}$ given a set of parameters $\boldsymbol{\theta}$:

$$\log p(\mathbf{x}_{1:T} | \boldsymbol{\theta}) \quad = \quad \sum_{t=1}^{T} \log p(\mathbf{x}_t | \mathbf{x}_{1:t-1}, \boldsymbol{\theta}). \tag{7.2}$$

[1] Note that we use the term ARMA to include vector ARMA, with no restriction to scalar time series.

Despite the fact that the conditional expectation $E[\mathbf{x}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\theta}]$ can be computed efficiently, the quantity $\log p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\theta})$ is not concave in $\boldsymbol{\theta}$ (Mauricio, 1995; Lütkepohl, 2007, Sec. 12.2-3), which suggests that maximizing the marginal likelihood is a hard computational problem. Another source of difficulty is that ARMA is a latent variable model, hence marginalizing over the unobserved innovations $\boldsymbol{\varepsilon}_{1:T}$ might also be problematic. Given innovations $\boldsymbol{\varepsilon}_{1:T} = [\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_T]$, however, $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\varepsilon}_{1:t-1}, \boldsymbol{\theta})$ is a simple Gaussian with mean

$$\boldsymbol{\mu}_t \quad = \sum_{i=1}^{p} A^{(i)} \mathbf{x}_{t-i} + \sum_{j=1}^{q} B^{(j)} \boldsymbol{\varepsilon}_{t-j} \tag{7.3}$$

and covariance Σ. To obtain such a simplified form, we will first characterize the entire data likelihood in terms of the innovations which enables application of the widely used expectation-maximization algorithm. This result parallels Lemma 1; however, we repeat it for this non-i.i.d. setting for clarity.

Lemma 4. *For an auxiliary density $q(\cdot)$ over $\boldsymbol{\varepsilon}_{1:T}$, and entropy $H(q(\cdot))$, it follows that (proof given in Appendix G.1):*

$$\log p(\mathbf{x}_{1:T}|\boldsymbol{\theta}) = \log \int p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\boldsymbol{\theta})\, d\boldsymbol{\varepsilon}_{1:T} = \max_{q(\cdot)} \int q(\boldsymbol{\varepsilon}_{1:T}) \log p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\boldsymbol{\theta})\, d\boldsymbol{\varepsilon}_{1:T} + H(q(\cdot)). \tag{7.4}$$

The maximum likelihood problem can now be re-expressed as $\min_{\boldsymbol{\theta}} \min_{\{q(\cdot)\}} -\log p(\mathbf{x}_{1:T}|\boldsymbol{\theta})$ where in a standard EM algorithm, the M step would consist of optimizing $\boldsymbol{\theta}$ given $\{q(\cdot)\}$, and the E step would consist of (implicitly) optimizing $\{q(\cdot)\}$ given $\boldsymbol{\theta}$ (Neal and Hinton, 1998). A standard variant of the log likelihood in (7.4) can then be obtained simply by dropping the entropy regularizer $H(q(\cdot))$. This leads to the minimization selecting a Dirac delta distribution on $\boldsymbol{\varepsilon}_{1:T}$ and a far simpler formulation, sometimes known as "hard EM" or "Viterbi EM" (Brown et al., 1993):

$$\begin{aligned} &\min_{\boldsymbol{\theta}} \min_{\boldsymbol{\varepsilon}_{1:T}} -\log p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\boldsymbol{\theta}) \\ &= \min_{\boldsymbol{\theta}} \min_{\boldsymbol{\varepsilon}_{1:T}} -\sum_{t=1}^{T} \Big[\log p(\boldsymbol{\varepsilon}_t|\mathbf{x}_{1:t}, \boldsymbol{\varepsilon}_{1:t-1}, \boldsymbol{\theta}) + \log p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\varepsilon}_{1:t-1}, \boldsymbol{\theta}) \Big]. \end{aligned} \tag{7.5}$$

This formulation suggests an approach where one successively imputes values $\boldsymbol{\varepsilon}_{1:T}$ for the unobserved innovation variables, then optimizes the parameters. Interestingly, $p(\boldsymbol{\varepsilon}_t|\mathbf{x}_{1:t}, \boldsymbol{\varepsilon}_{1:T-1}, \boldsymbol{\theta})$ is a Dirac delta distribution; $\boldsymbol{\varepsilon}_t$ must be the residual, $\boldsymbol{\varepsilon}_t = \mathbf{x}_t - \boldsymbol{\mu}_t$, otherwise the loss becomes unbounded. This distribution, therefore, imposes a constraint on the minimization. To maximize

likelihood under this constraint, we optimize

$$\min_{\substack{\boldsymbol{\theta},\boldsymbol{\varepsilon}_{1:T}:\\ \boldsymbol{\varepsilon}_t=\mathbf{x}_t-\boldsymbol{\mu}_t}} -\sum_{t=1}^{T}\log p(\mathbf{x}_t|,\mathbf{x}_{1:t-1},\boldsymbol{\varepsilon}_{1:t-1},\boldsymbol{\theta}) \qquad \triangleright\, \boldsymbol{\mu}_t=\sum_{i=1}^{p}A^{(i)}\mathbf{x}_{t-i}+\sum_{j=1}^{q}B^{(j)}\boldsymbol{\varepsilon}_{t-j} \tag{7.6}$$

$$= \min_{\substack{\boldsymbol{\theta},\boldsymbol{\varepsilon}_{1:T}:\\ \boldsymbol{\varepsilon}_t=\mathbf{x}_t-\boldsymbol{\mu}_t}} \frac{T}{2}\log((2\pi)^n|\Sigma|)+\frac{1}{2}\sum_{t=1}^{T}\left\|\Sigma^{-\frac{1}{2}}(\mathbf{x}_t-\boldsymbol{\mu}_t)\right\|^2. \tag{7.7}$$

Unfortunately, this optimization raises an important challenge. Due to the direct interaction between Σ, B and $\boldsymbol{\varepsilon}_{1:T}$ the final form of the problem (7.7) is still not convex in the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varepsilon}_{1:T}$ jointly. A typical strategy is therefore to first estimate the innovations $\boldsymbol{\varepsilon}_{1:T}$ directly from data, for example by using the errors of a learned autoregressive model, then observing that with the innovation variables fixed and Σ approximated from the innovations, the problem becomes convex in $\boldsymbol{\theta}$ (Hannan and Kavalieris, 1984; Lütkepohl, 2007). Another more contemporary approach is to convert the ARMA model into a state-space model (see Figure 7.1(b)) and then solve for parameters in that model using system identification approaches (Bauer, 2005). Though this has been an important advance for efficient ARMA estimation, these approaches still result in local minima.

7.2 Regularized ARMA modeling

To develop a likelihood based criterion that admits efficient global optimization, we begin by considering a number of extensions to the ARMA model. First, notice that the ARMA model in (7.1) can be equivalently formulated by introducing a $B^{(0)}$ and taking $\boldsymbol{\varepsilon}_t \sim \mathcal{N}(\boldsymbol{\mu}=\mathbf{0},\Sigma=I)$, giving $B^{(0)}\boldsymbol{\varepsilon}_t \sim \mathcal{N}(\mathbf{0},B^{(0)}B^{(0)'})$.

Second, following (Thiesson et al., 2012) an independent noise term $\boldsymbol{\eta}_t$ can be added to (7.1) to obtain the *stochastic ARMA* model

$$\mathbf{x}_t=\sum_{i=1}^{p}A^{(i)}\mathbf{x}_{t-i}+\sum_{j=0}^{q}B^{(j)}\boldsymbol{\varepsilon}_{t-j}+\boldsymbol{\eta}_t. \tag{7.8}$$

A key challenge in estimating the parameters of a classical ARMA model (7.1) is coping with the deterministic constraint that $\boldsymbol{\eta}_t = 0$, which forces the innovations $\boldsymbol{\varepsilon}_t$ to match the residuals (7.7). The stochastic ARMA model (7.8) relaxes this assumption by allowing $\boldsymbol{\eta}_t$ to be generated by a smooth exponential family distribution, such as $\boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0},Q_t)$ for covariance Q_t; a smaller Q_t yields a closer approximation to the original ARMA model. Thiesson et al. (2012) have shown that, for $Q_t=\sigma I$, expectation-maximization (EM) updates are only meaningful for $\sigma > 0$; for $\sigma = 0$, EM stops after one iteration. EM is not however guaranteed to find a globally optimal parameter estimate for the stochastic ARMA model. A key advantage of this model, however, is that it allows

a convenient re-expression of the marginal log-likelihood (7.6) by applying the chain rule in the opposite order for $\mathbf{x}_t$ and $\boldsymbol{\varepsilon}_t$:

$$\begin{aligned}(7.6) &= \min_{\boldsymbol{\theta}} \min_{\boldsymbol{\varepsilon}_{1:T}} -\sum_{t=1}^{T} \Big[\log p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\varepsilon}_{1:t}, \boldsymbol{\theta}) + \log p(\boldsymbol{\varepsilon}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\varepsilon}_{1:t-1}, \boldsymbol{\theta})\Big] \\ &= \min_{\boldsymbol{\theta}} \min_{\boldsymbol{\varepsilon}_{1:T}} -\sum_{t=1}^{T} \Big[\log p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\varepsilon}_{1:t}, \boldsymbol{\theta}) + \log p(\boldsymbol{\varepsilon}_t|\boldsymbol{\theta})\Big],\end{aligned}$$

since $\boldsymbol{\varepsilon}_t$ is independent of past innovations and data without $\mathbf{x}_t$. Furthermore, $p(\boldsymbol{\varepsilon}_t|\boldsymbol{\theta}) = p(\boldsymbol{\varepsilon}_t)$ since the covariance was moved into $B^{(0)}$ to make $\boldsymbol{\varepsilon}_t \sim \mathcal{N}(\mathbf{0}, I)$, yielding

$$-\sum_{t=1}^{T} \log p(\boldsymbol{\varepsilon}_t) = \frac{nT}{2}\log(2\pi) + \frac{1}{2}\sum_{t=1}^{T} \|\boldsymbol{\varepsilon}_t\|_2^2 = \frac{nT}{2}\log(2\pi) + \frac{1}{2}\|\mathcal{E}\|_F^2 \tag{7.9}$$

for $\mathcal{E} = \boldsymbol{\varepsilon}_{1:T}$. The constant is ignored in the optimization.

Third, rather than merely consider a maximum likelihood objective, we can consider the maximum a posteriori (MAP) estimate given by the introduction of a prior $\log p(\boldsymbol{\theta})$ over the model parameters $\boldsymbol{\theta} = (A, B, \Sigma)$. Since the parameters A and B do not typically have distributional assumptions, we view the choice of priors rather as regularizers:

$$-\log p(\boldsymbol{\theta}) = -\log p(A) - \log p(B) = R(A) + G(B),$$

for convex functions R and G. Any convex regularizer on A is acceptable. The choice of regularizer on B is more subtle, since for any s, $B\mathcal{E} = (Bs^{-1})(s\mathcal{E})$: $G(B)$ is required to prevent B from being scaled up, pushing $\|\mathcal{E}\|_F^2$ to zero. We consider $G(B) = \|B\|_F^2$ for $B = [B^{(0)}; \ldots; B^{(q)}]$, which effectively controls the size of B and, importantly, also results in a global reformulation given in Theorem 9.

Finally, as noted, we can consider any natural exponential family distribution for $\boldsymbol{\eta}_t$ rather than merely assuming Gaussian. The negative log-likelihood for such a distribution corresponds to a regular Bregman divergence (see Appendix G.3), allowing one to write the final estimation criterion in terms of a convex loss function $L(\cdot|\mathbf{x}_t)$ as

$$\min_{A,B,\mathcal{E}} \sum_{t=1}^{T} L\Big(\sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i} + \sum_{j=0}^{q} B^{(j)}\boldsymbol{\varepsilon}_{t-j} \;\Big|\; \mathbf{x}_t\Big) + \alpha\Big(\|\mathcal{E}\|_F^2 + G(B)\Big) + \gamma R(A), \tag{7.10}$$

for regularization parameters α and γ.

7.3 Efficient parameter estimation

Although L is convex in its first argument, the regularized criterion (7.10) is not jointly convex due to the coupling between B and $\mathcal{E}$. However, using the insights from Chapter 4, we can reformulate (7.10) as a convex optimization.

For fixed autoregressive parameters, A, let $L_{A,t}(\mathbf{z}) = L(\mathbf{z} + \sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i} \mid \mathbf{x}_t) + \gamma R(A)$, which is still convex in $\mathbf{z}$.[2] By introducing the change of variables, $Z = B\mathcal{E}$, the optimization over B and $\mathcal{E}$ given A can be written as

$$\min_{A,B,\mathcal{E}} \sum_{t=1}^{T} L_{A,t}\Big(\sum_{j=0}^{q} B^{(j)}\mathcal{E}_{:,t-j}\Big) + \alpha\Big(\|\mathcal{E}\|_F^2 + G(B)\Big) \tag{7.11}$$

$$= \min_{A,Z} \sum_{t=1}^{T} L_{A,t}\Big(\sum_{j=0}^{q} Z^{(j)}_{:,t-j}\Big) + \alpha \min_{\substack{B,\mathcal{E} \\ B\mathcal{E}=Z}} \Big(\|\mathcal{E}\|_F^2 + G(B)\Big). \tag{7.12}$$

This objective can be re-expressed in a convex form since

$$\|\!|Z|\!\| = \min_{B,\mathcal{E}:B\mathcal{E}=Z} \Big(\|\mathcal{E}\|_F^2 + G(B)\Big) \tag{7.13}$$

defines an induced norm on Z for two settings of $G(B)$ described in Theorems 9 and 10 below, which rely on Theorem 3, Theorem 7 and Corollary 3 from Chapter 4. Therefore, (7.12) can be equivalently expressed as:[3]

$$\sum_{t=1}^{T} L\Big(\sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i} + \sum_{j=0}^{q} Z^{(j)}_{:,t-j} \;\Big|\; \mathbf{x}_t\Big) + \alpha\|\!|Z|\!\| + \gamma R(A).$$

We can alternate between A and Z to obtain a globally optimal solution, then recover B and $\mathcal{E}$ from Z. Proofs for the following two theorems are provided in Appendix G.2.

Theorem 9. *The regularized ARMA(p, q) estimation problem for $G(B) = \|B\|_F^2$ is equivalent to*

$$(7.11) = \min_{A,Z} \sum_{t=1}^{T} L_{A,t}\Big(\sum_{j=0}^{q} Z^{(j)}_{:,t-j}\Big) + 2\alpha \|Z\|_{tr}$$

with a singular value decomposition recovery: $Z = U\Sigma V'$ giving $B = U\sqrt{\Sigma}$ and $\mathcal{E} = \sqrt{\Sigma}V'$.

The estimation problem is more difficult with the second choice of regularizer; to get an exact formulation, we need to restrict $q = 1$, giving $Z = \begin{bmatrix} B^{(0)} \\ B^{(1)} \end{bmatrix}\mathcal{E}$.

Theorem 10. *The regularized ARMA(p, 1) estimation problem for $G(B) = \max_{j=0,\ldots,q} \big\|B^{(j)}\big\|_F^2$ is equivalent to*

$$(7.11) = \min_{A,Z} \sum_{t=1}^{T} L_{A,t}\Big(\sum_{j=0}^{q} Z^{(j)}_{:,t-j}\Big) + \max_{0\le\eta\le 1} \Big\|E_\eta^{-1}Z\Big\|_{tr} \tag{7.14}$$

where $E_\eta := \begin{bmatrix} 1/\sqrt{\eta}\, I_n & 0 \\ 0 & 1/\sqrt{1-\eta}\, I_n \end{bmatrix}$. Moreover $\big\|E_\eta^{-1}Z\big\|_{tr}$ is concave in η over $[0, 1]$, enabling an efficient line-search.

[2] Proved in Lemma 12, Appendix G.2 for completeness

[3] Proved in Corollary 6, Appendix G.2 for completeness.

7.4 Identifiability and optimal parameter recovery

One desirable ideal for estimation is identifiability: being able to "identify" parameters uniquely. For a strictly convex loss function, L, the convex regularized ARMA optimization in (7.14) produces a unique moving average variable, Z. This identifiable matrix is sufficient for correctly estimating the autoregressive parameters for the ARMA model, which can be all that is required for forecasting in expectation.

It might be desirable, however, to recover the factors B and $\mathcal{E}$ to gain further insight into the nature of the time series. Unfortunately, unlike Z, the factors that satisfy $B\mathcal{E} = Z$ are not unique. Worse, if one simply recovers any B and $\mathcal{E}$ that satisfies $B\mathcal{E} = Z$, the recovered innovations $\mathcal{E}$ need not be Gaussian distributed. This issue can be addressed via a careful recovery procedure that finds a particular pair B and $\mathcal{E}$ with the same regularization penalty as Z. Let

$$\text{Factors}(Z) = \left\{ (B, \mathcal{E}) : B\mathcal{E} = Z \text{ and } G(B) + \|\mathcal{E}\|_F^2 = \|\!|Z|\!\| \right\}$$

This set of solutions satisfies the desired distributional properties, but is invariant under scaling and orthogonal transformations: for any $(B, \mathcal{E}) \in \text{Factors}(Z)$, (i) for $s = G(B)/\|\mathcal{E}\|_F$, $(B(1/s), s\mathcal{E}) \in \text{Factors}(Z)$ and (ii) for any orthogonal matrix $P \in \mathbb{R}^{k\times k}$, $(BP, P'\mathcal{E}) \in \text{Factors}(Z)$ since the Frobenius norm is invariant under orthogonal transformations. When $G(B) = \|\mathcal{E}\|_F$, a solution from Factors(Z) can be computed from the singular value decomposition of Z, as shown in Theorem 9. For $G(B) = \max_j \left\|B^{(j)}\right\|_F^2$, the boosting procedure in Algorithm 2 with rescaling Algorithm 6 can be used; see Appendix G.3 for a discussion on obtaining Laplacian instead of Gaussian innovations, which involves instead using a $(2,1)$-block norm on $\mathcal{E}$ and Algorithm 5 for rescaling.

7.5 Computational complexity

The overall estimation procedure is outlined in Algorithm 13 for the simpler regularizer, $\|B\|_F^2$; the approach is similar for the other regularizer, but with an outer line search over η. The computational complexity is governed by the matrix multiplication to compute the autoregressive and moving average components, and by the use of the singular value decomposition. The matrix multiplication for AX_p is $O(Tpn^2)$, which dominates the cost of computing the autoregressive loss, corresponding to ARloss in Algorithm 13. For the moving average loss, MALoss in Algorithm 13, the thin SVD of $Z \in \mathbb{R}^{qn\times T}$ has complexity $O(Tq^2n^2)$ and the multiplication of UV' is also $O(Tq^2n^2)$. Thus, each call to ARloss is $O(Tpn^2)$ and each call to MAloss is $O(Tq^2n^2)$. The initial solution of A, which involves solving a basic vector autoregressive model, is $i_1 O(Tpn^2)$ where i_1 is the number of iterations which is typically small. For i_2 the number of iterations then between A and B: RARMA

Algorithm 13 RARMA(p, q)

Input: X, p, q, α, γ
Output: $A, B, \mathcal{E}$

$X_p =$ history(X, p) = $[X_{p:t}; \ldots; X_{1:t-p}]$
[f,g] = MAloss(Z, A):
    $[U, \Sigma, V] =$ svd(Z)
    $Y = AX_p + \sum_{j=1}^{q} Z(j : n(j+1) - 1, :)$
    f $= L(Y; X) + \alpha$ sum(diag(Σ))
    g = repmat($\nabla_Y L(Y; X)$, q, 1)
    // Zero out unused parts of Z
    for $j = 1, \ldots q$, g$(j\!:\!n(j\!+\!1)$, $(t\!-\!j\!+\!1)\!:\!t) = 0$
    g = g $+\alpha UV'$
[f,g] = ARloss(A, Z):
    $Y = AX_p + \sum_{j=1}^{q} Z_{j:n(j+1)-1,:}$
    $f = L(Y; X) + \alpha R(A)$
    $g = (\nabla_Y L(Y; X))X_p + \gamma \nabla R(A)$
Initialize $Z = \mathbf{0}, A = \mathbf{0}$
// Apply your favourite optimizer to the AR search
$A =$ lbfgs(ARloss($\cdot, Z$), A)
// Iterate between A and Z
$[A, Z]$ = iterate(ARloss, MAloss, A, Z)
// Recover the optimal B and $\mathcal{E}$
$[U, \Sigma, V] =$ svd(Z)
$B = U\Sigma^{1/2}$, $\mathcal{E} = \Sigma^{1/2}V'$

Return: $A, B, \mathcal{E}$

cost = VAR cost + $i_2(O(Tpn^2)+O(Tq^2n^2)) = (i_1+i_2)O(Tpn^2)+i_2O(Tq^2n^2) \approx O(T(p+q^2)n^2)$.

7.6 Experimental evaluation

In this section, regularized ARMA is compared to a wide range of time series methods for the task of forecasting future observations on both synthetic and real-world data. As discussed in Appendix G.4.3, forecasts are performed using available innovations for the first q steps, and afterwards only the autoregressive parameters. In the final section, there is also a comparison on estimating the underlying autoregressive parameters of RARMA using $q = 0$ versus $q > 0$.

Several algorithms are compared: MEAN, which uses the mean of the training sequence as the prediction, a popular subspace identification method (N4SID) (Van Overschee and De Moor, 1994), expectation-maximization to learn the parameters for a Kalman filter (EM-Kalman), Hilbert space embeddings of hidden Markov models (HSE-HMM) (Song et al., 2010; Boots and Gordon, 2012) [4], maximum likelihood estimation of vector AR (AR), the Hannan-Rissanen method for

[4] In addition to the two method-of-moments approaches, N4SID and HSE-HMM, we tried a third state-space technique (Anandkumar et al., 2012), with no previous published empirical demonstrations. It performed poorly and so is

Table 7.1: For each dataset, the first column contains the test MSE (with standard error in parentheses) and the second the percentage of trials that were stable. The stability is evaluated using a threshold: eigenvalues $< 1+\epsilon = 1.01$. The method(s) with the most t-test wins with significance level of 5% is(are) bold for each dataset. Stable rates are key for iterated prediction performance; large MSE is mainly due to unstable trials.

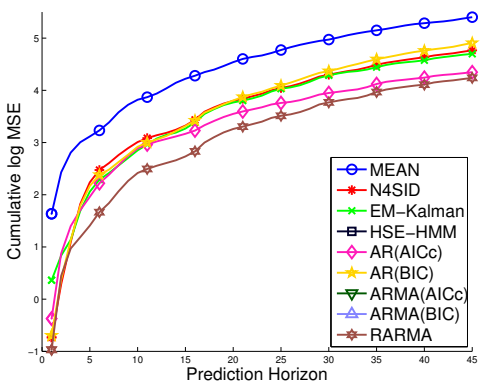
ALGORITHM	N6-P2-Q2		N9-P3-Q3		N12-P3-Q3		N15-P3-Q3		N12-P4-Q4	
MEAN	2.85(0.13)	1.00	6.64(0.27)	1.00	12.5(0.44)	1.00	21.2(1.06)	1.00	6.81(0.19)	1.00
N4SID	3.23(0.21)	1.00	6.82(0.3)	1.00	12.9(0.58)	1.00	24.7(1.46)	1.00	6.85(0.19)	1.00
EM-KALMAN	4.27(0.3)	0.97	11.7(1.17)	0.94	19(1.19)	0.95	32.8(3.03)	0.89	15.9(1.45)	0.88
HSE-HMM	13.5(12.8)	0.95	1070(1469)	0.95	353(514)	0.95	2017(3290)	0.95	31.8(28.6)	0.91
AR(AICC)	1.83(0.29)	0.96	8.99(2.38)	0.88	16.9(2.69)	0.84	80.2(24.7)	0.69	24.8(29.6)	0.69
AR(BIC)	1.67(0.25)	0.97	6.42(1.27)	0.91	10.7(1.22)	0.93	34(5.2)	0.85	5.25(0.54)	0.82
ARMA(AICC)	1.63(0.2)	0.98	4.93(0.62)	0.93	8.52(0.73)	0.96	27.7(3.64)	0.86	5.49(0.48)	0.88
ARMA(BIC)	1.68(0.25)	0.98	4.81(0.58)	0.95	8.4(0.59)	0.97	29.4(5.76)	0.91	5.26(0.48)	0.97
RARMA	**1.29**(0.08)	1.00	**4.1**(0.3)	0.97	**7.49**(0.41)	0.99	**15.7**(0.92)	0.98	**4.69**(0.21)	0.99

Table 7.2: As in Table 7.1, test MSE and stability are reported, now on two real datasets.

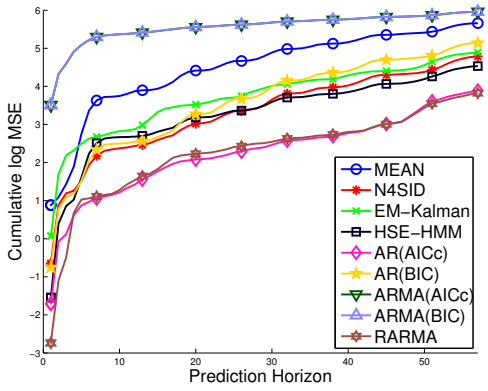
ALGORITHM	CAC		ATLANTIC	
MEAN	4.92(0.17)	1.00	5.03(0.39)	1.00
N4SID	2.6(0.35)	1.00	2.1(0.98)	1.00
EM KALMAN	2.43(0.26)	1.00	2.33(0.66)	1.00
HSE-HMM	7637(1433)	0.88	1.63(0.53)	1.00
AR(AICC)	**1.71**(0.31)	1.00	**0.85**(0.37)	1.00
AR(BIC)	3.00(0.27)	1.00	2.99(0.81)	1.00
ARMA(AICC)	101(42.9)	1.00	6.8(2.85)	1.00
ARMA(BIC)	107(49.3)	1.00	6.8(2.85)	1.00
RARMA	**1.53**(0.27)	1.00	**0.8**(0.35)	1.00

ARMA (ARMA-HR) (Hannan and Kavalieris, 1984) and global estimation of regularized ARMA (RARMA). We also compared to local alternation of the RARMA objective (7.11); for both the best and worst random initializations, however, the results were always worse that global RARMA, slower by more than an order of magnitude and often produced unstable results. Therefore, these local alternator results are omitted, with the focus instead on the comparison between the RARMA objective and the other approaches. The HR method is used for the ARMA implementation, because a recent study (Kascha, 2012) shows that HR is reasonably stable compared to other vector ARMA learning algorithms. A third step was added for further stability, in which the $A^{(i)}$ are re-learned from the observations with the MA component (from the second step) removed. The built-in Matlab

omitted.



(a) CAC.



(b) Atlantic.

Figure 7.2: Cumulative test MSE in log scale on two real-world datasets. Each model is iterated for (a) 40 and (b) 60 steps, respectively. AR (BIC) appears to perform a strong 1-step prediction, but then quickly degrades in performance, indicating the importance in selecting a good lag length for AR. HSE-HMM is unstable for CAC, but performs reasonably for Atlantic. The best performing methods are N4SID, AR(AICc), and RARMA. In both, RARMA has the lowest error for predictions up to a horizon of 30.

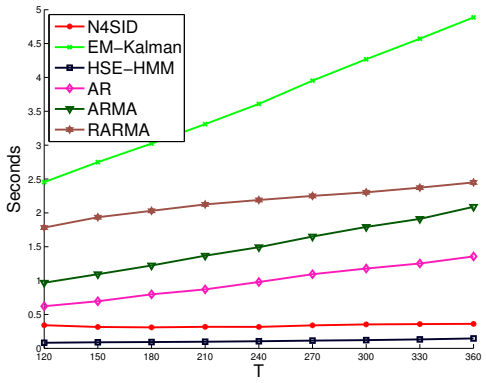


Figure 7.3: Runtimes for an increasing number of samples, where the parameters of the multivariate series are $n = 9$ and $p = q = 3$. The method-of-moments approaches are the fastest and appear to be less affected by increasing number of samples. The maximum likelihood approaches, however, are comparable, with EM-Kalman affected most by the increase in samples. Interestingly, the gap between RARMA and ARMA decreases as samples increase, and remains parallel with the simpler AR implementation.

implementations were used for AR and N4SID.

The lag parameters p and q were selected according to standard criteria in time series. For AR and ARMA-HR, the parameters p and q are chosen using BIC and AICc, and reported separately due to interesting differences in their performance. For N4SID, the built-in Matlab implementation chose the best order. For RARMA, because of the temporal structure in the data, parameters were chosen by performing estimation on the first 90% of the training sequence and evaluating model performance on the last 10% of the training sequence. Euclidean loss is used as the loss function L for RARMA, to make it more comparable to the other methods; other (robust) losses would likely give a greater advantage and are easily incorporated due to the generality of RARMA.

7.6.1 Synthetic experiments

Synthetic data sets are generated from an ARMA(p, q) model. An ARMA model is called unstable if the spectra of the AR matrices exceeds the unit circle on the complex plane (Lütkepohl, 2007); intuitively, iterating a dynamics matrix $A = U\Sigma V'$ that has any $\Sigma(i, i) > 1$ for t steps, $A^t = U\Sigma^t V'$, is expansive. See Appendix G.4.1 for details about the procedure for generating stable ARMA models. For each (p, q, n) configuration, 100 data sequences are generated, each with 300 samples partitioned into 200 training points followed by 100 test points.

Table 7.1 shows the results, including the test mean squared error (MSE) and the stability rates over all trials. RARMA with global optimization is among the best models on each data set in terms of MSE. Learning is generally more difficult as the dimension increases, but RARMA performs well even when most algorithms fail to beat the baseline (MEAN) and maintains a reasonably stable rate. Figure 7.3 illustrates a runtime comparison, in CPU seconds. The synthetic model is fixed to $n = 9$ and $p = q = 3$, with an increasing number of training points. RARMA is comparable to other methods in terms of efficiency and scales well with increasing number of samples T.

7.6.2 Experiments on real time series

To see how our method performs on real-world data, we experimented on two real-world datasets from IRI [5]. These climate datasets consist of monthly sea surface temperature on the tropical Pacific Ocean (CAC) and the tropical Atlantic Ocean (Atlantic). The area size of CAC is 84×30 with 399 points, while the area of Atlantic is 22×11 with 564 points. We use the first 30×30 locations for CAC and the first 9×9 locations for Atlantic. These areas are further partitioned into grids of size 3×3 and vectorized to obtain observations $\mathbf{x}_t \in \mathbb{R}^9$. The data for each location is normalized to have sample mean of zero and sample standard deviation of one in the experiments. The first 90% of the sequence is used as training set, the last 10% as the test set.

Table 7.1 shows the test MSE and Figure 7.2 shows the cumulative MSE in log scale. As in the synthetic experiments, RARMA is consistently among the best models in terms of MSE. Moreover, when examining iterated forecasting accuracy in Figure 7.2, RARMA is better for early predictions (about the first 30 predictions) on the real datasets.

7.6.3 Investigating the moving average component

The final comparison is an investigation into the importance of the moving average component, versus simply using an AR model. RARMA(p, q) is compared to RARMA$(p, 0)$ and RARMA$(p+q, 0)$ for two settings: recovering the true autoregressive parameters, A, and accuracy in forecasting. The

[5] http://iridl.ldeo.columbia.edu/SOURCES/

same code is run for all three methods, simply with different p, q settings. The comparison is over increasing lag and increasing variance of the moving average component. The results are presented in Figure 7.4. The heat map presents the relative error between RARMA(p, q) and both RARMA($p, 0$) and RARMA($p+q, 0$); values greater than 1 indicate superior performance for RARMA(p, q).

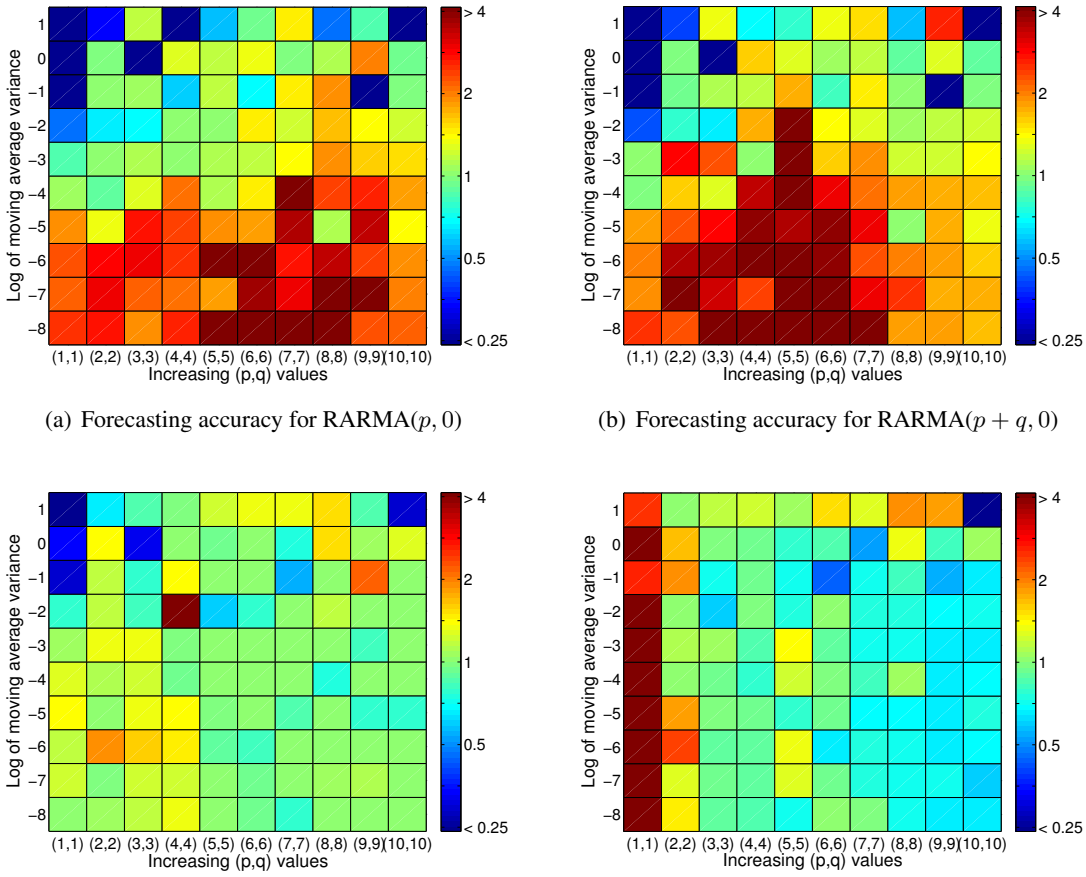
These results indicate three interesting phenomena. First, including the moving average component significantly improves performance. As the variance reached levels where the mean began to outperform all three techniques, the models with $q=0$ were slightly more stable. Second, RARMA with $q > 0$ performs noticeably better for forecasting but typically performed about the same or worse for extracting the underlying autoregressive parameters. This result is counter to the expectation that a more accurate A directly results in improved forecasting performance and suggests that, if the ultimate goal is forecasting, we need not focus so strongly on identification as is typically done in empirical work for ARMA. Interestingly, as p, q increase, RARMA($p + q$,0) becomes the dominant method for obtaining the underlying parameters, despite the fact that only the first p components of the learned A are used. Finally, often when RARMA($p, 0$) performed poorly in forecasting, RARMA($p+q, 0$) performed well, and vice versa. The fact that RARMA(p, q) performed comparably to the best of the two suggests that that the moving average component improves robustness. Importantly, because vector ARMA models can now be solved globally, there is little disadvantage in using this more powerful model, and strong benefits in some cases.

7.7 Summary

This chapter tackled a long-standing problem in time series modeling: tractable maximum likelihood estimation of multivariate ARMA models. The approach involves three key components: (1) estimating stochastic ARMA models, which relaxes the requirement that the innovations exactly equal the residuals, (2) characterizing the independent Gaussian structure of the innovations using a regularizer and (3) developing a theoretically sound convex reformulation of the resulting stochastic multivariate ARMA objective. Solving this convex optimization is efficient, guarantees global solutions and outperformed previous ARMA and state-space methods in forecasting on synthetic and real datasets.

These results suggest stochastic regularized ARMA is a promising direction for time series modeling, over conventional (deterministic) ARMA. Stochastic ARMA is similarly motivated by the Wold representation, but is amenable to optimization, unlike deterministic ARMA. Moreover, the regularized ARMA objective facilitates development of estimation algorithms under generalized innovations. Though the focus in this work was on a convex formulation for Gaussian innovations, it extends to Laplacian innovations for a $(2, 1)$-block norm regularizer. Advances in optimizing

structured norm objectives could result in important advances in global estimation of regularized ARMA models for novel innovation properties.



(a) Forecasting accuracy for RARMA$(p, 0)$

(b) Forecasting accuracy for RARMA$(p + q, 0)$

(c) Parameter recovery for RARMA$(p, 0)$

(d) Parameter recovery for RARMA$(p + q, 0)$

Figure 7.4: The relative error is reported between RARMA(p, q) and the RARMA$(p,0)$ and RARMA$(p + q,0)$: err(RARMA$(q = 0)$) / err(RARMA$(q > 0)$). The plot is symmetric, where at 4, RARMA(p, q) has 4x lower error (good), and at 0.25, has 4x higher error (bad). The dimension is set to $n = p$ and $50 + p + q$ training samples. The x-axis shows increasing lag and the y-axis increasing moving average variance. As the variance is increased beyond $\exp(-3)$, using the mean as a predictor begins to outperform all three methods. For **(a)** and **(b)**, the comparison is with respect to forecasting accuracy for a horizon of 10, measured with ℓ_1 error. For **(c)** and **(d)**, the comparison is with respect to the ℓ_1 error between the recovered A parameters and the true parameters, cut-off at p for RARMA$(p + q, 0)$. Interestingly, it appears that the accuracy of A is not crucial for forecasting performance, as RARMA(p, q) outperforms RARMA$(p, 0)$ for most reasonable innovation variance in terms of forecasting error, but not in terms of accuracy of the underlying A.

Chapter 8

Perspectives and future work

This dissertation set out to address the question

> What *unifications* and *algorithmic development* for statistical machine learning problems can be achieved under regularized factor models?

This dissertation only provides a partial answer to such an ambitious question. In terms of unification, a modest understanding has been added to a growing literature that seeks to unify problems and connect algorithms. The most novel additions are formulating semi-supervised learning as two different regularized factorization and formulating estimation of autoregressive moving average models as a regularized factorization. For algorithmic development, the focus has been on using regularization to encode structure and convex reformulations to obtain global solutions. Combining the novel insights developed in this dissertation with previous work, the resulting partial answer is that regularized factor models (1) greatly simplify the interpretation of many machine learning algorithms as the maximum likelihood estimation of a factorization of the data, (2) but are still simple enough to simultaneously facilitate algorithmic development, such as in the form of convex reformulations.

In the long-term, however, I hope to say something stronger about using these models for promoting precise problem specification and for directing algorithmic development. For example, there is a clear connection between the number of clusters for hard or soft clustering, the latent dimension in dimensionality reduction and the hidden state dimension in time series models; these connections can give insight into specifying this problem property. Other model classes have been crucial for algorithm development, such as semidefinite programming and linear programming. Regularized factor models (RFMs) could be a class of models for the machine learning community that enable similar benefits, but with a greater focus on unification.

This view may be ambitious, but I believe that it is an apt time to explore such general formalisms that successfully balance both optimization and generalization. This class deviates from

previous optimization formalisms in that only a subset of problems in the class can be currently solved. The expectation, however, is that we can slowly make progress on solving others, while maintaining useful connections. It moves away from the extremes of usual formalisms: either the problem specification has known solutions (e.g., linear programs) or favours universality (e.g., graphical models or maximum likelihood). The area in-between these extremes may have been less explored because it is conceptually more difficult to deal with this lack of precisely defined optimization properties. I believe, however, that if we can find a balance between these two extremes, it will be beneficial for the advancement of algorithmic development in machine learning and for reducing the barrier to entry for those outside the community.

8.1 Research directions

There are many avenues to explore for regularized factor models, particularly in terms of representational power, computational challenges and theoretical questions.

8.1.1 What cannot be represented as a regularized factor model?

Though an impressive number of problems can be reformulated as a regularized factorization, there are many important settings not covered by these models. In this section, I discuss a few of these settings, with a focus on those that are promising extensions to regularized factor models.

The most similar, but importantly different, missing formulation are *probabilistic approaches*, such as probabilistic PCA and factor analysis. Recall that a typical goal in single-view problems is to learn C and Φ such that $X_{:,t} = \mathbf{f}(C\Phi_{:,t}) + \boldsymbol{\eta}_t$, where the noise $\boldsymbol{\eta}_t$ is characterized by the chosen exponential family. Probabilistic approaches learn both C, Φ and the parameters for the distribution on the noise, such as the covariance R in the case $\boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, R)$. Probabilistic PCA, factor analysis and gaussian mixture models have all been shown to be an instance of probabilistic, Gaussian regularized factor models (Roweis and Ghahramani, 1999). For example, factor analysis corresponds to $X_{:,t} = C\Phi_{:,t} + \boldsymbol{\eta}_t$, $\boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, R)$, such that R is diagonal. Though this appears to add another parameter to the problem, there is in fact a clear connection to the regularization parameter. Consider probabilistic PCA, where the covariance is simply assumed to be a fixed variance for each dimension, σ^2, giving the maximum likelihood solution $\Phi C = U(\Sigma - \sigma^2 I)V'$ for $X = U\Sigma V'$ (Tipping and Bishop, 1999). This solution corresponds to the solution of the minimization $\frac{1}{2}\left\|X - C\Phi\right\|_F^2 + \sigma^2 \left\|C\Phi\right\|_{tr}$ [Theorem 2.1](Cai et al., 2010). Adding an outer minimization over the regularization parameter in the regularized factorization optimization, therefore, would result in probabilistic PCA. Conversely, the maximum likelihood solution for σ (Tipping and Bishop, 1999, Equation 8) could be used to set the regularization parameter. This direction, therefore, is

useful both for meta-parameter selection and incorporating a broader class of problems.

Tensor factorizations are another potential generalization for regularized factor models. A matrix is a 2-D tensor, encoding number of samples cross number of attributes; a tensor generalizes this to multiple dimensions, enabling multiple views, locations and time slices of data as well as other modes found in realistic scenarios, such as trials, task-conditions and subjects. Tensor decompositions and factorizations are not new, having been introduced in 1927 and used quite extensively in psychometric and chemometrics (Cichocki et al., 2009). In addition to this empirical understanding, a recent focus in computational fields, like signal processing and machine learning, has elucidated important algorithmic questions to advance the practicality of these methods on large modern datasets and so make it a potentially practical extension. Unfortunately, many problems that are feasible for matrices have been shown to be NP-hard for tensors (Hillar and Lim, 2013); to extend to tensors, therefore, it seems likely that a feasible avenue is to explore approximate solutions.

In addition to the "can we do it" question, we might also ask "is it really necessary" to extend regularized factor models to tensors. Matrices are much simpler to deal with than tensors. Moreover, results comparing single-view and multi-view learning in Section 5.4 indicated that in many situations, single-view learning performed almost as well as multi-view learning, despite the fact that structure information was ignored. Despite these facts, tensors have clear advantages. First, tensors are more compact than matrices. For example, unfolding a $10 \times 10 \times 10$ tensor creates a 1000×10 matrix: algorithms that reduce tensor dimensions much more quickly reduce the dataset size than those reducing dimension on the unfolded matrix. Second, algorithmic development encoding structure through constraints is limited. The multi-view work in Section 5.3 emulated a 3-D tensor, with the third dimension restricted to two views, using constraints on the matrix to create two partitions. This approach, however, is cumbersome and difficult to scale to higher order structure. In fact, we may find that higher-order structure is required to see significant improvements over the flat, single-view concatenation.

Finally, regularized factor models do not explicitly model hierarchical structure, such as is the case for (deep) neural networks. A single layer in a neural network, which consists of the mapping from a hidden layer to output, is a regularized factorization: $\mathbf{f}(C\Phi) \approx Y$. A one-layer neural network could be seen as a two-view regularized factorization, as in reduced rank regression or semi-supervised learning. One view corresponds to the input, X, the other to the output, Y, with hidden layer Φ; the model learned for X, however, is in terms of reverse prediction rather than forward prediction. It is not obvious, however, how to extend beyond one layer and maintain a class amenable to optimization. Better understanding how to solve a simple one-layer neural network, which likely involves further insights from the forward-reverse prediction framework, is an impor-

tant step for understanding if regularized factor models can be reasonably extended to include a hierarchy.

8.1.2 Computational challenges

A major theme of this thesis was a focus on obtaining convex reformulations, to ensure both efficient solvers and global solutions. There are, however, many remaining open questions about improving computational complexity and obtaining global guarantees for new settings.

Developing incremental algorithms for regularized factor models is crucial for making these approaches applicable to datasets of modern sizes. Though many of the solvers were an efficient minimization of a regularized loss, the algorithms presented are all inherently batch algorithms. As the data size T grows, the algorithm complexity on each step grows linearly in T. An incremental algorithm that processes one sample at a time avoids this issue, instead having computational complexity per step dependent only on the dimension of the data. The overall runtime to process all samples will still depend on T, but we avoid the need to fit all data into memory and perform expensive gradient computations on all samples. Bousquet and Bottou (2008) have shown both empirically and theoretically that stochastic gradient algorithms (incrementally processing data) can more quickly reach a solution with low expected cost than second-order batch methods. In addition to speed, incremental methods can also reduce the risk of overfitting.

Another important direction is to better understand the subset of regularized factor models that can be convexly reformulated. This direction requires more insight into the relationship between the chosen regularizers on C and Φ and the resulting induced norm on $Z = C\Phi$. Several of the reformulations presented in this work relied on relatively simply knowledge of dual norms; the partitioned reformulations, however, were quite complex. In general, there is no obvious way to obtain a closed form to the minimization $\|\!|Z|\!\| = \min_{\Phi,C} \sum_{i=1}^{\infty} \|C_{:,i}\|_C^2 + \|\Phi_{i,:}\|_R^2$ for any column and row regularizers $\|\cdot\|_C$ and $\|\cdot\|_R$. A numerical approach for obtaining the induced norm, or some approximation of it, would make these reformulations much more widely applicable. A numerical approach would also make it simpler to incorporate constraints on Φ and C, which complicate the derivation of a closed-form induced norm. If exact, closed-form solutions are desired, product distributions may provide a more intuitive recipe for obtaining the induced norm. Product distribution characterize the distribution of the product of two random variables. Because the regularizers on C and Φ specify the distribution on those variables, there may be a way to characterize the distribution of their product, $Z = C\Phi$, and so obtain regularizer $R(Z) = -\log p(Z)$.

Another important avenue is to explore what regularized factorization can be solved globally, even if we do not have a convex reformulation. Zhang et al. (2012) introduced a boosting approach

to optimizing regularized factor models, that iteratively generates rows and columns of C and Φ, similar to the boosting recovery algorithm in Algorithm 2. This boosting algorithm directly learns C and Φ, rather than Z, and is guaranteed to find a global solution under reasonable requirements. This approach, however, still requires an oracle to determine a descent direction, which appears to be tied to knowledge of the induced norm on Z and so does not expand the set of globally solvable regularized factorization. In certain settings, local alternation is guaranteed to find a global solution under reasonable initialization, such as in low-rank matrix completion (Jain et al., 2013; Jain and Dhillon, 2013) and in sparse coding (Agarwal et al., 2013). This work was inspired by the empirical success of local alternation in practice, such as the effectiveness of matrix completion for the Netflix challenge and wide-spread use of sparse coding for image and speech analysis. For the case where a global solver is known, global convergence of the local alternator was empirically confirmed in a trace-norm regularized alignment score model objective (Mirzazadeh et al., 2014). Better understanding the convergence properties of alternation in biconvex objectives, and further exploring biconvex optimization strategies (cf. (Gorski et al., 2007, Section 4.2)), could better elucidate new global optimization techniques for regularized factor models.

8.1.3 Theoretical challenges

An important question is understanding the theoretical utility of these factorized representations. Though in practice unsupervised learning has been shown to be useful, there is little theoretical understanding on the impacts of these new representations on prediction performance. There are several hypotheses, such as the *manifold hypothesis*, which states that the data is concentrated on a lower-dimensional manifold, and the *predictive representation hypothesis*, which states that representations using predictions about possible future experience give good generalization. There is some theoretical work on "learning to learn" or bias learning based on generalization properties of the learned representation. For example, the general principle behind multi-task learning is that the sample complexity for prediction is lower when training and using a shared representation (Baxter, 1995). Some classes of representations are known to have important representability properties. Two-layer neural networks (for a sufficiently large hidden layer) can represent any function (Barron, 1993; Andoni et al., 2014). Fourier basis functions can be used to represent any square-integrable function. Taylor series can be used to approximate functions that are infinitely differentiable in a neighbourhood around a point. Kernel-based approaches rely on the fact that any function in a reproducing kernel Hilbert space can be represented as an (infinite) weighted sum of kernel functions using Mercer's theorem; or, more practically, on the representer theorem stating that the function in the hypothesis class that minimizes empirical risk can be represented as the weighted sum of kernel

distances on the training points (Argyriou et al., 2009). To the best of my knowledge, however, there are no such theoretical results on using subspace representations. A theoretical investigation into if PCA improves sample complexity for a predictor is a plausible first step in understanding the usefulness of factorized representations.

The convergence properties of problems with two interacting parameters has been much less explored than for the single parameter setting, such as in supervised learning. Gribonval et al. (2013) recently characterized the sample complexity of various matrix factorization problems, as a factor of $\sqrt{\frac{\log T}{T}}$. For the setting where we can solve this matrix factorization, including those described in Chapter 4, this result correspondingly indicates consistency as $T \to \infty$. This analysis, however, applies only to the Euclidean loss setting; an interesting next step is to extend these finite sample error bounds to general convex losses.

The connection between forward and reverse prediction enabled important algorithmic insights for semi-supervised learning, but remains imprecise. The main issue is that for any forward regularizer, any reverse regularizer can be chosen and a forward-reverse equivalence obtained. Currently, the connection is obtained under hard EM, where the entropy regularizer is dropped for both the forward and reverse problems (see Appendix B.2). Because the entropy regularizer affects the choice of prior on the forward and reverse variables, this approximation likely results in this lack of precision on the specification of regularizers. Understanding the effects of the entropy regularizer appears to be the key component for better elucidating the connection between forward and reverse prediction. As mentioned earlier, this understanding is also key for understanding how to formalize a one-layer neural network as a regularized factor model.

8.2 Summary

This dissertation develops global solution techniques for regularized factor models and incorporates novel problem setting under regularized factor models. The central component of the algorithmic development is a general convex reformulation procedure that involves characterizing an induced norm on the joint variable, and then recovering the original factors either in closed form or with a boosting procedure. These algorithmic developments are then used to advance estimation in subspace learning, semi-supervised learning and auto-regressive moving average models, with empirical demonstrations of the advantages of the resulting simple algorithms. The results in this thesis further advocate for regularized factor models as a unified formalism for both future algorithm development and problem specification.

Bibliography

Agarwal, A., Anandkumar, A., Jain, P., and Netrapalli, P. (2013). Learning sparsely used overcomplete dictionaries via alternating minimization. *arXiv.org*, 1310.7991v2.

Akaho, S., Kiuchi, Y., and Umeyama, S. (1999). MICA: multimodal independent component analysis. In *Proceedings of the International Joint Conference on Neural Networks*, pages 927–932.

Anandkumar, A., Hsu, D., and Kakade, S. M. (2012). A method of moments for mixture models and hidden Markov models. *arXiv.org*, 1203.0683v3.

Anava, O., Hazan, E., Mannor, S., and Shamir, O. (2013). Online learning for time series prediction. *arXiv.org*, 1302.6927v1.

Andersson, S. (2009). Subspace estimation and prediction methods for hidden Markov models. *The Annals of Statistics*, pages 4131–4152.

Andoni, A., Panigrahy, R., Valiant, G., and Zhang, L. (2014). Learning polynomials with neural networks. *Proceedings of the 31st International Conference on Machine Learning*, pages 1908–1916.

Archambeau, C. and Bach, F. (2008). Sparse probabilistic projections. In *Advances in Neural Information Processing Systems*, pages 73–80.

Argyriou, A., Evgeniou, T., and Pontil, M. (2008). Convex multi-task feature learning. *Machine Learning*, 73(3):243–272.

Argyriou, A., Micchelli, C., and Pontil, M. (2009). When is there a representer theorem? Vector versus matrix regularizers. *Journal of Machine Learning Research*, 10:2507–2529.

Arslan, O. (2010). An alternative multivariate skew Laplace distribution: properties and estimation. *Statistical Papers*, pages 865–887.

Bach, F. and Jordan, M. I. (2006). A probabilistic interpretation of canonical correlation analysis. Technical Report 688, Berkeley.

Bach, F., Mairal, J., and Ponce, J. (2008). Convex sparse matrix factorizations. *arXiv.org*, 0812.1869v1.

Balcan, M.-F. and Blum, A. (2005). A PAC-style model for learning from labeled and unlabeled data. In *Proceedings of the 18th Annual Conference on Learning Theory (COLT)*, pages 111–126.

Banerjee, A., Merugu, S., Dhillon, I. S., and Ghosh, J. (2005). Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945.

Bauer, D. (2005). Estimating linear dynamical systems using subspace methods. *Econometric Theory*, 21(01).

Baxter, J. (1995). Learning internal representations. In *Proceedings of the International Conference on Learning Theory*, pages 311–320, New York, New York, USA. ACM Press.

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.

Belkin, M., Niyogi, P., and Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning*.

Ben-David, S., Lu, T., and Pal, D. (2008). Does unlabeled data provably help? Worst-case analysis of the sample complexity of semi-supervised learning. In *Proceedings of the 21st Annual Conference on Learning Theory*, pages 33–44.

Benveniste, A., Metivier, M., and Priouret, P. (2012). *Adaptive Algorithms and Stochastic Approximations*. Springer.

Bishop, C. M. (2006). Pattern recognition and machine learning. *Springer*.

Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the llth Annual Conference on Learning Theory*, pages 92–100.

Boots, B. and Gordon, G. (2012). Two-manifold problems with applications to nonlinear system identification. In *Proceedings of the 29th International Conference on Machine Learning*, pages 623–630.

Borga, M., Landelius, T., and Knutsson, H. (1997). A unified approach to PCA, PLS, MLR and CCA. Technical report, Linköping University.

Bousquet, O. and Bottou, L. (2008). The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems*, pages 161–168.

Boyd, S. P. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Bradley, D. M. and Bagnell, J. A. (2008). Differential sparse coding. In *Advances in Neural Information Processing Systems*.

Bradley, D. M. and Bagnell, J. A. (2009). Convex coding. In *Uncertainty and Control*, pages 83–90. AUAI Press.

Brand, M. (2003). Continuous nonlinear dimensionality reduction by kernel eigenmaps. In *Proceedings of the 18th International Joint Conference on Artifical Intelligence*, pages 547–554.

Brockwell, P. J. and Davis, R. A. (2002). *Introduction to Time Series and Forecasting*. Springer.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

Buntine, W. and Jakulin, A. (2006). Discrete Component Analysis. In *Subspace, Latent Structure and Feature Selection*, pages 1–33. Springer Berlin Heidelberg, Berlin, Heidelberg.

Cai, J.-F., Candès, E. J., and Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982.

Candes, E. J., Li, X., Ma, Y., and Wright, J. (2011). Robust principal component analysis? *Journal of the ACM*, 58:1–37.

Candes, E. J. and Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772.

Chen, H. and Peng, J. (2008). 0-1 semidefinite programming for graph-cut clustering: modelling and approximation. *Data mining and mathematical programming*, 45:15–40.

Cheng, H., Zhang, X., and Schuurmans, D. (2013). Convex relaxations of Bregman divergence clustering. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, pages 162–171.

Cichocki, A., Zdunek, R., Phan, A. H., and Amari, S. (2009). *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley.

Collins, M., Dasgupta, S., and Schapire, R. E. (2002). A generalization of principal component analysis to the exponential family. In *Advances in Neural Information Processing Systems*, pages 617–624.

Comon, P. (1994). Independent component analysis, a new concept? *Signal Processing*, 36(3):287–314.

Corduneanu, A. and Jaakkola, T. (2006). Data dependent regularization. *Semi-Supervised Learning*, pages 163–182.

Cortes, C. and Mohri, M. (2007). On transductive regression. In *Advances in Neural Information Processing Systems*, pages 305–312.

Cramér, H. (1946). *Mathematical Methods of Statistics*. Princeton University Press.

Cristianini, N. (2003). Convex methods for transduction. In *Advances in Neural Information Processing Systems*, pages 73–80.

De Bie, T., Cristianini, N., and Rosipal, R. (2005). Eigenproblems in pattern recognition. In *Handbook of Geometric Computing: Applications in Pattern Recognition, Computer Vision, Neuralcomputing, and Robotics*, pages 129–170.

De Bie, T. and De Moor, B. (2002). On two classes of alternatives to canonical correlation analysis, using mutual information and oblique projections. In *Proceedings of the 23rd Symposium on Information Theory*.

De la Torre, F. (2012). A least-squares framework for component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1041–1055.

Dhillon, I., Guan, Y., and Kulis, B. (2004). Kernel k-means: spectral clustering and normalized cuts. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 551–556.

Dhillon, P. S., Foster, D., and Ungar, L. (2011). Multi-view learning of word embeddings via CCA. In *Advances in Neural Information Processing Systems*, pages 199–207.

Ding, C., Li, T., and Peng, W. (2006). Nonnegative matrix factorization and probabilistic latent semantic indexing: equivalence, chi-square statistic, and a hybrid method. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 342–348.

Druck, G. and McCallum, A. (2010). High-performance semi-supervised learning using discriminatively constrained generative models. In *Proceedings of the 27th International Conference on Machine Learning*, pages 319–326.

Durbin, J. (1960). The fitting of time-series models. *Revue de l'Institut International de Statistique*, 28:233–244.

Ehm, W., Genton, M. G., and Gneiting, T. (2003). Stationary covariances associated with exponentially convex functions. *Bernoulli*, 9(4):607–615.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745.

Eustaquio, R. G. and Karas, E. W. (2008). Constraint qualifications for nonlinear programming. *Federal University of Parana*.

Fisher, R. A. (1938). The statistical utilization of multiple measurements. *Annals of Eugenics*, 8(4):376–386.

Foster, D. P., Rodu, J., and Ungar, L. H. (2012). Spectral dimensionality reduction for HMMs. *arXiv.org*.

Gaussier, E. and Goutte, C. (2005). Relation between PLSA and NMF and implications. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 601–602.

Geiger, D., Verma, T., and Pearl, J. (1990). Identifying independence in Bayesian networks. *Networks*, 20(5):507–534.

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

Goldberg, A., Zhu, X., Recht, B., and Xu, J. (2010). Transduction with matrix completion: Three birds with one stone. In *Advances in Neural Information Processing Systems*, pages 757–765.

Goldfarb, D., Ma, S., and Scheinberg, K. (2010). Fast alternating linearization methods for minimizing the sum of two convex functions. *SIAM Journal on Optimization*.

Gordon, G. (2003). Generalized^2 Linear^2 Models. In *Advances in Neural Information Processing Systems*, pages 593–600.

Gorski, J., Pfeuffer, F., and Klamroth, K. (2007). Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical methods of operations research*, pages 373–407.

Gribonval, R., Jenatton, R., Bach, F., Kleinsteuber, M., and Seibert, M. (2013). Sample complexity of dictionary learning and other matrix factorizations. *arXiv.org*, 1312.3790v1.

Ham, J., Lee, D. D., Mika, S., and Schölkopf, B. (2004). A kernel view of the dimensionality reduction of manifolds. In *Proceedings of the 21st International Conference on Machine Learning*, pages 47–54.

Hannan, E. J. and Kavalieris, L. (1984). Multivariate linear time series models. *Advances in Applied Probability*, 16:492–561.

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

He, X. and Niyogi, P. (2004). Locality preserving projections. In *Advances in Neural Information Processing Systems*, pages 153–160.

Hendrickx, J. M. and Olshevsky, A. (2010). Matrix p-norms are NP-hard to approximate if $p \neq 1, 2, \infty$. *SIAM Journal on Matrix Analysis and Applications*, 31(5):2802–2812.

Hillar, C. J. and Lim, L.-H. (2013). Most tensor problems are NP-hard. *Journal of the ACM*, 60(6):1–39.

Hinton, G. E. (2007). Learning multiple layers of representation. *Trends in Cognitive Sciences*, 11(10):428–434.

Hochreiter, S. and Schmidhuber, S. (1999). Source separation as a by-product of regularization. In *Advances in Neural Information Processing Systems*, pages 459–465.

Horn, R. A. and Johnson, C. R. (1990). *Matrix Analysis*. Cambridge University Press.

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

Hsu, D., Kakade, S., and Zhang, T. (2012). A spectral algorithm for learning Hidden Markov Models. *Journal of Computer and System Sciences*, pages 1460–1480.

Hyvärinen, A., Hurri, J., and Hoyer, P. O. (2009). *Natural Image Statistics*. A probabilistic approach to early computational vision. Springer, London.

Jain, P. and Dhillon, I. S. (2013). Provable inductive matrix completion. *arXiv.org*, 1306.0626v1.

Jain, P., Netrapalli, P., and Sanghavi, S. (2013). Low-rank matrix completion using alternating minimization. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 665–674, New York, New York, USA. ACM Press.

Jenatton, R., Mairal, J., Obozinski, G., and Bach, F. (2010). Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the 27th International Conference on Machine Learning*, pages 487–494.

Jia, Y., Salzmann, M., and Darrell, T. (2010). Factorized latent spaces with structured sparsity. In *Advances in Neural Information Processing Systems*, pages 982–990.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning*, pages 200–209.

Jong, J. C. and Kotz, S. (1999). On a relation between principal components and regression analysis. *The American Statistician*, 53(4):349–351.

Journée, M., Bach, F., Absil, P. A., and Sepulchre, R. (2010). Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimzation*, 20(5):2327–2351.

Kascha, C. (2012). A comparison of estimation methods for vector Autoregressive Moving-Average Models. *Econometric Reviews*, 31(3):297–324.

Katayama, T. (2006). *Subspace Methods for System Identification*. Springer.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82–95.

Kivinen, J. and Warmuth, M. K. (2001). Relative loss bounds for multidimensional regression problems. *Journal of Machine Learning Research*, 45(3):301–329.

Kokiopoulou, E., Chen, J., and Saad, Y. (2011). Trace optimization and eigenproblems in dimension reduction methods. *Numerical Linear Algebra with Applications*, pages 565–602.

Kulis, B., Dhillon, I., and Mooney, R. (2009). Semi-supervised graph clustering: a kernel approach. *Journal of Machine Learning Research*, 74:1–22.

Lampert, C. H. and Kromer, O. (2010). Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning. In *European Conference on Computer Vision*, pages 566–579.

Lee, H., Raina, R., Teichman, A., and Ng, A. (2009). Exponential family sparse coding with applications to self-taught learning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1113–1119.

Lee, J. A. and Verleysen, M. (2010). *Nonlinear Dimensionality Reduction*. Springer.

Li, Y. and Zhou, Z. (2011). Towards making unlabeled data never hurt. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1081–1088.

Liski, E., Nordhausen, K., and Oja, H. (2013). Supervised invariant coordinate selection. *Statistics*, pages 1–21.

Long, B., Yu, P. S., and Zhang, Z. M. (2008). A general model for multiple view unsupervised learning. In *Proceedings of the SIAM International Conference on Data Mining*, pages 822–833.

Lütkepohl, H. (2007). *New Introduction to Multiple Time Series Analysis*. Springer.

Ma, S., Goldfarb, D., and Chen, L. (2011). Fixed point and Bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128:321–353.

Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., and Bach, F. R. (2008). Supervised dictionary learning. In *Advances in Neural Information Processing Systems*, pages 1033–1040.

Mauricio, J. A. (1995). Exact maximum likelihood estimation of stationary vector ARMA models. *Journal of the American Statistical Association*, pages 282–291.

Mirzazadeh, F., Guo, Y., and Schuurmans, D. (2014). Convex co-embedding. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 1989–1996.

Moonen, M. and Ramos, J. (1993). A subspace algorithm for balanced state space system identification. *IEEE Transactions on Automatic Control*, 38(11):1727–1729.

Murata, N., Takenouchi, T., Kanamori, T., and Eguchi, S. (2004). Information geometry of U-Boost and Bregman divergence. *Neural Computation*, 16(7):1437–1481.

Nadler, B., Srebro, N., and Zhou, X. (2009). Semi-supervised learning with the graph laplacian: the limit of infinite unlabelled data. *Advances in Neural Information Processing Systems*, pages 1330–1338.

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

Nigam, K. and Ghani, R. (2000). Analyzing the effectiveness and applicability of co-training. In *Proceedings of the 9th International Conference on Information and Knowledge Management*, pages 86–93.

Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Journal of Machine Learning*, 39:103–134.

Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, 37(23):3311–3325.

Overton, M. L. and Womersley, R. S. (1993). Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices. *Mathematical Programming*, 62(1-3):321–357.

Pataki, G. (1998). On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues . *Mathematics of Operations Research*, 23(2):339–358.

Peng, J. and Wei, Y. (2007). Approximating K-means- type clustering via semidefinite programming. *SIAM Journal on Optimization*, pages 186–205.

Pereira, F. and Gordon, G. (2006). The support vector decomposition machine. In *Proceedings of the 23rd Internal Conference in Machine Learning*, pages 689–696, New York, New York, USA. ACM Press.

Petersen, K. B. (2004). The matrix cookbook. *Technical University of Denmark*.

Petz, D. (1994). A survey of certain trace inequalities. *Functional analysis and operator theory*, pages 287–298.

Phatak, A. and de Hoog, F. (2002). Exploiting the connection between PLS, Lanczos methods and conjugate gradients: alternative proofs of some properties of PLS. *Journal of Chemometrics*, 16(7):361–367.

Quadrianto, N. and Lampert, C. H. (2011). Learning multi-view neighborhood preserving projections . In *Proceedings of the 28th International Conference on Machine Learning*, pages 425–432.

Recht, B., Fazel, M., and Parrilo, P. A. (2010). Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *Siam Review*, 52(3):471–501.

Redner, R. A. and Walker, H. F. (1984). Mixture densities, maximum likelihood and the EM algorithm. *Siam Review*, 26(2):195–239.

Rish, I., Grabarnik, G., Cecchi, G., Pereira, F., and Gordon, G. J. (2008). Closed-form supervised dimensionality reduction with generalized linear models. In *Proceedings of the 25th international conference on Machine learning*, pages 832–839, New York, New York, USA. ACM Press.

Rockafellar, R. (1970). *Convex Analysis*. Princeton U. Press.

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

Roweis, S. and Ghahramani, Z. (1999). A unifying review of linear Gaussian models. *Neural Computation*, 11(2):305–345.

Salakhutdinov, R. and Srebro, N. (2010). Collaborative filtering in a non-uniform world: Learning with the weighted trace norm. In *Advances in Neural Information Processing Systems*, pages 2056–2064.

Scargle, J. D. (1981). Studies in astronomical time series analysis. *The Astrophysical Journal Supplement Series*, pages 835–853.

Schölkopf, B. and Smola, A. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press.

Schölkopf, B., Smola, A., and Müller, K.-R. (1997). Kernel principal component analysis. In *Artificial Neural Networks—ICANN'97*, pages 583–588. Springer-Verlag, Berlin/Heidelberg.

Shah, P., Bhaskar, B. N., Tang, G., and Recht, B. (2012). Linear System Identification via Atomic Norm Regularization. *arXiv.org*, 1204.0590v1.

Shalev-Shwartz, S., Srebro, N., and Zhang, T. (2010). Trading accuracy for sparsity in optimization problems with sparsity constraints. *SIAM Journal on Optimization*, pages 2807–2832.

Sigal, L., Memisevic, R., and Fleet, D. J. (2009). Shared kernel information embedding for discriminative inference . In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2852–2859.

Sindhwani, V., Niyogi, P., and Belkin, M. (2005). Beyond the point cloud: from transductive to semi-supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 824–831.

Singh, A. P. and Gordon, G. J. (2008). A unified view of matrix factorization models. In *Machine Learning and Knowledge Discovery in Databases*, pages 358–373. Springer Berlin Heidelberg, Berlin, Heidelberg.

Smith, N. and Eisner, J. (2005). Contrastive estimation: training log-linear models on unlabeled data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 354–362.

Song, L., Boots, B., Siddiqi, S., Gordon, G., and Smola, A. (2010). Hilbert space embeddings of hidden Markov models. In *Proceedings of the 27th International Conference on Machine Learning*, pages 991–998.

Srebro, N., Rennie, J., and Jaakkola, T. (2004). Maximum-margin matrix factorization . In *Advances in Neural Information Processing Systems*, pages 1329–1336.

Steinberg, D. (2005). *Computation of matrix norms with applications to robust optimization*. PhD thesis, Technion.

Sun, L., Ji, S., and Ye, J. (2009a). A least squares formulation for a class of generalized eigenvalue problems in machine learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1–8. ACM Press.

Sun, L., Ji, S., and Ye, J. (2011). Canonical correlation analysis for multilabel classification: a least-squares formulation, extensions, and analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):194–200.

Sun, L., Ji, S., Yu, S., and Ye, J. (2009b). On the equivalence between canonical correlation analysis and orthonormalized partial least squares. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1230–1235.

Thiesson, B., Chickering, D. M., Heckerman, D., and Meek, C. (2012). ARMA time-series modeling with graphical models. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 552–560.

Tipping, M. E. and Bishop, C. M. (1999). Probabilistic principal component analysis. *Journal Royal Statistical Society B*, 61(3):611–622.

van den Wollenberg, A. L. (1977). Redundancy analysis an alternative for canonical correlation analysis. *Psychometrika*, 42(2):207–219.

van der Burg, E. and de Leeuw, J. (1990). Nonlinear redundancy analysis. *British Journal of Mathematical and Statistical Psychology*, pages 217–230.

van der Heijden, P. G. M., Gilula, Z., and van der Ark, L. A. (1999). An extended study into the relationship between correspondence analysis and latent class analysis. *Sociological Methodology*, 29:147–186.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning*, pages 2579–2605.

Van Overschee, P. and De Moor, B. (1994). N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1):75–93.

Viberg, M. (1995). Subspace-based methods for the identification of linear time-invariant systems. *Automatica*, 31(12):1835–1851.

Viinikanoja, J., Klami, A., and Kaski, S. (2010). Variational Bayesian mixture of robust CCA models. In *Machine Learning and Knowledge Discovery in Databases*, pages 370–385. Springer Berlin Heidelberg.

Wang, C. and Mahadevan, S. (2009). Manifold alignment without correspondence. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1273–1278.

Wegelin, J. A. (2000). A survey of Partial Least Squares (PLS) methods, with emphasis on the two-block case. Technical report.

Weinberger, K. Q. and Saul, L. K. (2006). Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, pages 77–90.

Weiss, Y. (1999). Segmentation using eigenvectors: a unifying view. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, pages 975–982.

White, M. and Schuurmans, D. (2012). Generalized optimal reverse prediction. In *Proceedings of the 15th International Conference on Artifical Intelligence and Statistics*, pages 1305–1313.

White, M., Wen, J., Bowling, M., and Schuurmans, D. (2015). Optimal estimation of multivariate ARMA models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, page To appear.

White, M., Yu, Y., Zhang, X., and Schuurmans, D. (2012). Convex multi-view subspace learning. In *Advances in Neural Information Processing Systems*, pages 1673–1681.

Wiesel, A., Bibi, O., and Globerson, A. (2013). Time varying autoregressive moving average models for covariance estimation. *IEEE Transactions on Signal Processing*, 61(11):2791–2801.

Williams, C. K. I. (2002). On a connection between kernel PCA and metric multidimensional scaling. *Machine Learning*, 46(1/3):11–19.

Wold, H. (1985). Partial least squares. *Encyclopedia of statistical sciences*.

Wold, H. O. A. (1938). *A Study in The Analysis of Stationary Time Series*. Almqvist & Wiskell.

Xing, E. P. and Jordan, M. I. (2003). On semidefinite relaxations for normalized k-cut and connections to spectral clustering. Technical Report UCB/CSD-03-1265, Berkeley.

Xu, H., Caramanis, C., and Sanghavi, S. (2010). Robust PCA via outlier pursuit. In *Advances in Neural Information Processing Systems*, pages 2496–2504.

Xu, L., White, M., and Schuurmans, D. (2009). Optimal reverse prediction: a unified perspective on supervised, unsupervised and semi-supervised learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1137–1144.

Yan, S., Xu, D., Zhang, B., Zhang, H.-J., Yang, Q., and Lin, S. (2007). Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40–51.

Zhang, X., Yu, Y., and Schuurmans, D. (2012). Accelerated training for matrix-norm regularization: A boosting approach. In *Advances in Neural Information Processing Systems*, pages 2906–2914.

Zhang, X., Yu, Y., White, M., Huang, R., and Schuurmans, D. (2011). Convex sparse coding, subspace learning, and semi-supervised extensions. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, pages 567–573.

Zhao, H. and Poupart, P. (2014). A sober look at spectral learning. *arXiv.org*, 1406.4631v1.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, pages 321–328.

Zhou, D. and Schölkopf, B. (2006). Discrete Regularization. In *Semi-Supervised Learning*, pages 237–249. MIT Press.

Zhu, X. (2006). Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison.

Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15:262–286.

Appendix A

Background information

A.1 Generalized eigenvalue problems

The eigenvalue decomposition for a square, diagonalizable matrix $A \in \mathbb{R}^{n \times n}$ is $A = Q\Lambda Q^{-1}$ giving $A\mathbf{q}_i = \lambda_i \mathbf{q}_i$ for $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_n]$. For a symmetric matrix A, Q is orthogonal, giving $A = Q\Lambda Q'$ with $QQ' = Q'Q = I$. This eigenvalue decomposition corresponds to the following optimization (Kokiopoulou et al., 2011):

$$\max_{V'V=I} \operatorname{tr}(V'AV) = \operatorname{tr}(Q'AQ) = \operatorname{tr}(\Lambda) = \sum_i \lambda_i.$$

The generalized eigenvalue problem is the solution to the equation $A\mathbf{q} = \lambda B\mathbf{q}$ for some matrix B. For a symmetric A, the generalized eigenvalue decomposition $A = BQ\Lambda Q'$ corresponds to

$$\max_{V'BV=I} \operatorname{tr}(V'AV) = \operatorname{tr}(Q'AQ) = \operatorname{tr}(\Lambda) = \sum_i \lambda_i.$$

Note that if B in the generalized eigenvalue problem is invertible, then we can always do a standard eigenvalue decomposition on $B^{-1}A$.

The singular value decomposition is related to the eigenvalue decomposition in that for the singular value decomposition $X = U\Sigma V'$ which has orthogonal U and V and diagonal Σ, the corresponding eigenvalue decomposition for $X'X$ and XX' respectively are

$$X'X = V\Sigma U'U\Sigma V' = V\Sigma^2 V'$$
$$XX' = U\Sigma V'V\Sigma U' = U\Sigma^2 U'$$

Only square, diagonalizable matrices have an eigenvalue decomposition; any matrix, however, has a singular value decomposition. Since any real, symmetric matrix is diagonalizable, we know that $X'X$ and XX' are both diagonalizable and so have an eigenvalue decomposition.

When minimizing rather than maximizing the above trace, the solution corresponds to the minimum eigenvalues of the generalized eigenvalue problem. For $B = I$, this minimization is also equivalent to obtaining the maximum eigenvalues of the pseudo-inverse matrix:

$$\min_{V'V=I} \operatorname{tr}(V'AV) = \max_{V'V=I} \operatorname{tr}(V'A^{\dagger}V)$$

This is because for singular value decomposition of a matrix $X = U\Sigma V'$, the pseudo-inverse has the decomposition $X^{\dagger} = V\Sigma^{-1}U'$, since it satisfies the four properties of a pseudo-inverse

1. $(XX^{\dagger})' = U\Sigma^{-1}V'V\Sigma U' = U\Sigma^{-1}\Sigma U' = U\Sigma\Sigma^{-1}U' = U\Sigma V'V\Sigma^{-1}U' = XX^{\dagger}$
2. $(X^{\dagger}X)' = V\Sigma U'U\Sigma^{-1}V' = V\Sigma^{-1}\Sigma V' = V\Sigma^{-1}U'U\Sigma V' = X^{\dagger}X$
3. $XX^{\dagger}X = U\Sigma V'V\Sigma^{-1}U'U\Sigma V' = U\Sigma\Sigma^{-1}\Sigma V' = U\Sigma V' = X$
4. $X^{\dagger}XX^{\dagger} = V\Sigma^{-1}U'U\Sigma V'V\Sigma^{-1}U' = V\Sigma^{-1}\Sigma\Sigma^{-1}U' = V\Sigma^{-1}U' = X^{\dagger}$

Therefore, the eigenvalue decomposition of $A^{\dagger}$ similarly has the inverse eigenvalues.

A.2 Relationship between regularization and constraints

In some settings, there are nice relationships between regularization and constraints. In the convex reformulations, we take advantage of these relationships. Often there is an advantage to enforcing constraints as regularizers, since constrained optimization can be more difficult. In this section, I quickly summarize these connections, to make the optimization choices in this thesis more clear.

A.2.1 Lagrangian, strong duality and the KKT conditions

In certain situations, we can take an equality and inequality constrained optimization and convert it into a regularized problem (see (Boyd and Vandenberghe, 2004, Chapter 5) for more detailed information). Assume the goal is to optimize

$$\min_{\mathbf{x}:\ \mathbf{g}(\mathbf{x})\leq\mathbf{0},\ \mathbf{h}(\mathbf{x})=\mathbf{0}} L(\mathbf{x})$$

where $L : \mathbb{R}^n \to \mathbb{R}$ is a loss function, $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^{c_g}$ is a vector-valued function of c_g inequality constraints and $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^{c_h}$ is a vector-valued function of c_h equality constraints. Notice that if $\mathbf{h}(\mathbf{x}) = \mathbf{c}$ for some vector, then we can define $\tilde{\mathbf{h}}(\mathbf{x}) = \mathbf{h}(\mathbf{x}) - \mathbf{c}$ and enforce $\tilde{\mathbf{h}}(\mathbf{x}) = 0$ and similarly for $\mathbf{g}$. Assume also that the constrained optimization has a solution, i.e., $\left(\bigcap_{i=1}^{c_g} \operatorname{dom} \mathbf{g}_i\right) \cap \left(\bigcap_{i=1}^{c_h} \operatorname{dom} \mathbf{h}_i\right) \cap \operatorname{dom} L$ is non-empty. Define the **Lagrange function**

$$\Lambda(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = L(\mathbf{x}) + \boldsymbol{\lambda}'\mathbf{g}(\mathbf{x}) + \boldsymbol{\nu}'\mathbf{h}(\mathbf{x}) = L(\mathbf{x}) + \sum_{i=1}^{c_g} \boldsymbol{\lambda}_i \mathbf{g}_i(\mathbf{x}) + \sum_{i=1}^{c_h} \boldsymbol{\nu}_i \mathbf{h}_i(\mathbf{x})$$

where $\boldsymbol{\lambda} \geq \mathbf{0}$ and $\boldsymbol{\nu}$ are called the *dual variables* or *Lagrange multiplier vectors* associated with the original constrained optimization. To obtain an equivalence, we first need to discuss the dual problem and strong duality.

The *Lagrange dual problem* associated with the original *primal problem* consists of the following optimization:

$$\max_{\boldsymbol{\lambda} \geq \mathbf{0},\, \boldsymbol{\nu}} R(\boldsymbol{\nu}, \boldsymbol{\lambda}) \qquad \text{where} \qquad R(\boldsymbol{\nu}, \boldsymbol{\lambda}) = \inf_{\mathbf{x}} \Lambda(\mathbf{x}, \boldsymbol{\nu}, \boldsymbol{\lambda})$$

Since the Lagrange dual function R gives a lower bound on the optimal value to the primal problem for each pair $(\boldsymbol{\nu}, \boldsymbol{\lambda})$, the dual optimization can be interpreted as finding the *best* lower bound for the primal problem using the Lagrange dual function. Interestingly, regardless of if L is convex, the dual R is concave in $\boldsymbol{\nu}$ and $\boldsymbol{\lambda}$, making the dual problem a convex optimization, since the objective is concave and the constraint is convex.

Using the max-min inequality, we automatically obtain the **weak-duality** property

$$\sup_{\boldsymbol{\lambda} \geq \mathbf{0},\, \boldsymbol{\nu}} R(\boldsymbol{\nu}, \boldsymbol{\lambda}) = \sup_{\boldsymbol{\lambda} \geq \mathbf{0},\, \boldsymbol{\nu}} \inf_{\mathbf{x}} \Lambda(\mathbf{x}, \boldsymbol{\nu}, \boldsymbol{\lambda}) \leq \inf_{\mathbf{x}} \sup_{\boldsymbol{\lambda} \geq \mathbf{0},\, \boldsymbol{\nu}} \Lambda(\mathbf{x}, \boldsymbol{\nu}, \boldsymbol{\lambda}) = \inf_{\mathbf{x}:\, \mathbf{g}(\mathbf{x}) \leq \mathbf{0},\, \mathbf{h}(\mathbf{x}) = \mathbf{0}} L(\mathbf{x}).$$

The optimization with the supremum and infimum swapped corresponds to the original primal problem because the inner supremum forces the outer optimization to choose $\mathbf{x}$ that satisfies the constraints; otherwise, the loss becomes unbounded.

To obtain **strong duality**, we need this inequality to be an equality which is often called the *strong max-min property* or the *saddle-point property*. This equality is obtained only under certain conditions, called *constraint qualifications*. An example of a widely used constraint qualification is *Slater's condition*, which requires that for convex L and linear equality constraints, $\mathbf{h}(\mathbf{x}) = A\mathbf{x} - \mathbf{c}$, there exists $\mathbf{x}$ such that $A\mathbf{x} = \mathbf{c}$ and $\mathbf{g}_i(\mathbf{x}) < 0$ for all $i = 1, \ldots, c_g$. This conditions equates to requiring that $\mathbf{x}$ is *strictly feasible*: the inequality constraints must be able to be satisfied as strict inequalities. If Slater's condition holds, then strong duality is obtained. For other constraint qualifications, including simplifications to Slater's condition under certain assumptions, see (Boyd and Vandenberghe, 2004, Section 5.2.3) and (Eustaquio and Karas, 2008).

When strong duality is known to hold for the problem, then an equivalence can be obtained between the constrained primal problem and the regularized (dual) problem. Let $\mathbf{x}^*$ and $(\boldsymbol{\nu}^*, \boldsymbol{\lambda}^*)$ be the primal and dual optimal points respectively, with zero duality gap (i.e., assuming that strong duality holds and so the optimal values for the primal and dual problems are equal). Therefore, the gradient of the Lagrangian must be zero at $\mathbf{x}^*$:

$$\nabla L(\mathbf{x}^*) + \sum_{i=1}^{c_g} \boldsymbol{\lambda}_i \nabla \mathbf{g}_i(\mathbf{x}^*) + \sum_{i=1}^{c_h} \boldsymbol{\nu}_i \nabla \mathbf{h}_i(\mathbf{x}^*) = 0$$

giving

$$\begin{aligned}
\mathbf{g}(\mathbf{x}^*) &\leq \mathbf{0} \\
\mathbf{h}(\mathbf{x}^*) &\leq \mathbf{0} \\
\boldsymbol{\lambda}^* &\geq \mathbf{0} \\
\forall i = 1, \ldots, c_g, \;\; \boldsymbol{\lambda}_i^* \mathbf{g}_i(\mathbf{x}^*) &= 0 \\
\nabla L(\mathbf{x}^*) + \sum_{i=1}^{c_g} \boldsymbol{\lambda}_i \nabla \mathbf{g}_i(\mathbf{x}^*) + \sum_{i=1}^{c_h} \boldsymbol{\nu}_i \nabla \mathbf{h}_i(\mathbf{x}^*) &= 0
\end{aligned}$$

which are called the **KKT conditions**. These KKT conditions can often be used to facilitate solving the primal problem, either by providing an analytic solution or by simplifying the optimization.

Therefore, given strong duality and a solution to the Lagrange function, we know that using the regularizer weights $(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$ gives a regularized problem that is equivalent to the original constrained optimization. Even under weak duality, we know that there are $(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$ that give the closest regularized problem to the original constrained problem. Without a characterization of the duality gap, however, it is difficult in this case to say exactly how the two problems relate.

A.2.2 General set constraints

Set constraints that place $\Phi \in \mathcal{F}$ or $C \in \mathcal{C}$ are not typically thought of as priors; to maintain the density interpretation, we turn to the Dirac delta distribution. There is not a well-defined density p for the Dirac delta, since it puts infinite weight on a set of measure zero. Instead, the Dirac delta is usually defined as the limit of the following function

$$\delta_a(x) = \frac{1}{a\sqrt{\pi}} \exp(-x^2/a^2).$$

If we take the negative logarithm of this function, and remove constants which would not be used in the minimization, we obtain $R_a(x) = \frac{x^2}{a^2}$. For $x = 0$, we get $R_a(x) \to 0$ as $a \to 0$. For any other x, we get $R_a(x) \to \infty$ as $a \to 0$. We can generalize this notion to any set using an indicator function, $\mathbb{1}_{\mathcal{F}} : \mathbb{R}^{k \times T} \to \{0, 1\}$:

$$\mathbb{1}_{\mathcal{F}}(\Phi) := \begin{cases} 1 & \text{if } \Phi \in \mathcal{F} \\ 0 & \text{otherwise} \end{cases}$$

Again, $R_{\Phi,k}(\Phi) \equiv -\log(\mathbb{1}_{\mathcal{F}}(\Phi)) = 0$ if $\Phi \in \mathcal{F}$ and otherwise is ∞. Therefore, the loss is dominated by the prior, unless the variables are in the pre-specified set.

Viewing hard set constraints as regularizers, however, does not appear to give any gains and simply obfuscates the problem. Therefore, we keep both constraint sets $\mathcal{C}$ and $\mathcal{F}$ and regularizers $R_{C,k}$ and $R_{\Phi,k}$ in the formulation of regularized factor models.

A.3 Bregman divergence properties

The following is a well-known property of Bregman divergences. It is useful to enable either D_F or $D_{F^\star}$ to be used, since Bregman divergences are only guaranteed to be convex in the first argument.

Lemma 2 With $\hat{\mathbf{y}} = \mathbf{f}(\hat{\mathbf{z}})$ and $\mathbf{y} = \mathbf{f}(\mathbf{z})$,

$$D_F(\hat{\mathbf{z}}||\mathbf{z}) = D_{F^\star}(\mathbf{y}||\hat{\mathbf{y}}). \tag{A.1}$$

Proof: Recall that $F^\star(\mathbf{y}) = \max_{\mathbf{z}} \mathbf{z}'\mathbf{y} - F(\mathbf{z})$, and solving for the maximum $\mathbf{z}$ we obtain

$$\frac{d}{d\mathbf{z}} = \mathbf{y} - \nabla F(\mathbf{z}) = \mathbf{y} - \mathbf{f}(\mathbf{z}) = 0 \implies \mathbf{z} = \mathbf{f}^{-1}(\mathbf{y})$$

giving

$$F^\star(\mathbf{y}) = \mathbf{f}^{-1}(\mathbf{y})'\mathbf{y} - F(\mathbf{f}^{-1}(\mathbf{y})) \tag{A.2}$$

Now notice that, for $H(f)$ the Hessian of f

$$\begin{aligned}
\nabla F^\star(\mathbf{y}) &= H(\mathbf{f}^{-1})(\mathbf{y})'\mathbf{y} + \mathbf{f}^{-1}(\mathbf{y}) - H(\mathbf{f}^{-1})(\mathbf{y})'\nabla F(\mathbf{f}^{-1}(\mathbf{y})) \\
&= \mathbf{f}^{-1}(\mathbf{y}) + H(\mathbf{f}^{-1})(\mathbf{y})'\mathbf{y} - H(\mathbf{f}^{-1})(\mathbf{y})'\mathbf{f}(\mathbf{f}^{-1}(\mathbf{y})) \\
&= \mathbf{f}^{-1}(\mathbf{y}) + H(\mathbf{f}^{-1})(\mathbf{y})'\mathbf{y} - H(\mathbf{f}^{-1})(\mathbf{y})'\mathbf{y} \\
&= \mathbf{f}^{-1}(\mathbf{y}) \\
&= \mathbf{f}^\star(\mathbf{y})
\end{aligned}$$

Now we can rewrite $D_{F^\star}(\mathbf{y}||\hat{\mathbf{y}})$ in terms of F and $\mathbf{f}^{-1} = \mathbf{f}^\star$

$$\begin{aligned}
D_{F^\star}(\mathbf{y}||\hat{\mathbf{y}}) &= F^\star(\mathbf{y}) - F^\star(\hat{\mathbf{y}}) - \mathbf{f}^\star(\hat{\mathbf{y}})'(\mathbf{y} - \hat{\mathbf{y}}) \\
&= \mathbf{f}^{-1}(\mathbf{y})'\mathbf{y} - F(\mathbf{f}^{-1}(\mathbf{y})) - \mathbf{f}^{-1}(\hat{\mathbf{y}})'\hat{\mathbf{y}} + F(\mathbf{f}^{-1}(\hat{\mathbf{y}})) - \mathbf{f}^{-1}(\hat{\mathbf{y}})'(\mathbf{y} - \hat{\mathbf{y}}) \\
&= \mathbf{f}^{-1}(\mathbf{y})'\mathbf{y} - F(\mathbf{f}^{-1}(\mathbf{y})) + F(\mathbf{f}^{-1}(\hat{\mathbf{y}})) - \mathbf{f}^{-1}(\hat{\mathbf{y}})'\mathbf{y}
\end{aligned}$$

Finally, recall that $\hat{\mathbf{y}} = \mathbf{f}(\hat{\mathbf{z}})$ and $\mathbf{y} = \mathbf{f}(\mathbf{z})$, giving us

$$\begin{aligned}
D_{F^\star}(\mathbf{y}||\hat{\mathbf{y}}) &= \mathbf{f}^{-1}(\mathbf{y})'\mathbf{y} - F(\mathbf{f}^{-1}(\mathbf{y})) + F(\mathbf{f}^{-1}(\hat{\mathbf{y}})) - \mathbf{f}^{-1}(\hat{\mathbf{y}})'\mathbf{y} \\
&= \mathbf{f}^{-1}(\mathbf{f}(\mathbf{z}))'\mathbf{f}(\mathbf{z}) - F(\mathbf{f}^{-1}(\mathbf{f}(\mathbf{z}))) + F(\mathbf{f}^{-1}(\mathbf{f}(\hat{\mathbf{z}}))) - \mathbf{f}^{-1}(\mathbf{f}(\hat{\mathbf{z}}))'\mathbf{f}(\mathbf{z}) \\
&= \mathbf{z}'\mathbf{f}(\mathbf{z}) - F(\mathbf{z}) + F(\hat{\mathbf{z}}) - \hat{\mathbf{z}}'\mathbf{f}(\mathbf{z}) \\
&= F(\hat{\mathbf{z}}) - F(\mathbf{z}) - \mathbf{f}(\mathbf{z})'(\hat{\mathbf{z}} - \mathbf{z}) \\
&= D_F(\hat{\mathbf{z}}||\mathbf{z})
\end{aligned}$$

■

EXPONENTIAL FAMILY	**DENSITY** $p(\mathbf{z}\|\boldsymbol{\theta})$ $= \exp(\langle \mathbf{z}, \boldsymbol{\theta} \rangle - F(\boldsymbol{\theta}))p_0(\mathbf{z})$	**TRANSFERS** $\mathbf{f}(\boldsymbol{\theta}) \approx \mathbf{z}$ **LOSS** $= \min_{\boldsymbol{\theta}} F(\boldsymbol{\theta}) - \mathbf{z}'\boldsymbol{\theta}$
BERNOULLI $z \in \{0,1\}$ BINARY OUTCOME $\theta \in [0,1],\ \theta = \ln \frac{q}{1-q}$ q PROBABILITY OF 1	$p(z\|q) = q^z(1-q)^{1-z}$ $\mu = q\ p(z\|\theta) = \frac{\exp(\theta z)}{1+\exp(\theta)} p_0(z)$	LOGISTIC LOSS $F(\theta) = \ln(1+\exp(\theta))$ $f(\theta) = (1+\exp(-\theta))^{-1}$ $f(\theta) = q$ SIGMOIDAL TRANSFER
BERNOULLI, n-D $z \in \{0,1\}^n$, $\theta \in [0,1]^n$	$p(\mathbf{z}\|\mathbf{q}) = \prod_i \mathbf{q}_i^{\mathbf{z}_i}(1-\mathbf{q}_i)^{1-\mathbf{z}_i}$ $p(\mathbf{z}\|\boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}'\mathbf{z})}{1+\exp(\boldsymbol{\theta})} p_0(\mathbf{z})$	$F(\theta) = \mathbf{1}'\ln(\mathbf{1}+\exp(\boldsymbol{\theta}))$ $f(\theta) = (1+\exp(-\boldsymbol{\theta}))^{-1}$
BINOMIAL $z \in \{0,\ldots,N\}$ # SUCCESSES IN N TRIALS $\theta \in [0,N],\ \theta = \ln \frac{q}{1-q}$	$p(z\|q) = \binom{N}{z} q^z(1-q)^{1-z}$ $\mu = Nq$ $p(z\|\theta) = \frac{\exp(\theta z)}{1+\exp(\theta)} p_0(z)$	$F(\theta) = N\ln(1+\exp(\theta))$ $f(\theta) = N(1+\exp(-\theta))^{-1}$ $f(\theta) = Nq$
EXPONENTIAL $z \in \mathbb{R}^+$ $\theta = -\lambda < 0$	$p(z\|\theta) = -\theta\exp(\theta z)$	ITAKURA-SAITO DISTANCE $F(\theta) = -\ln(-\theta)$ $f(\theta) = -\theta^{-1}$
GAMMA $k > 0$ SHAPE PARAMETER $\theta = -\beta < 0$ RATE PARAMETER $k = 1$ IS EXPONENTIAL DIST.	$p(z\|\theta) = -\theta^k \exp(z\theta)p_0(z)$ $p_0(z) = \Gamma(k)^{-1} z^{k-1}$ $\exp(-F(-\theta)) = -\theta^k$ MODELS WEIGHT TIMES	$F(\theta) = -k\ln(-\theta)$ $f(\theta) = -k\theta^{-1}$
GAUSSIAN FOR MEAN $\boldsymbol{\theta} = \boldsymbol{\mu}$ AND COVARIANCE Σ	$p(\mathbf{z}\|\boldsymbol{\theta}) = C\exp(-\frac{1}{2}MD(\boldsymbol{\theta}))$ WHERE $MD(\boldsymbol{\theta}) = (\mathbf{z}-\boldsymbol{\theta})'\Sigma^{-1}(\mathbf{z}-\boldsymbol{\theta})$ AND $C = (2\pi)^{-n/2}\lvert\Sigma\rvert^{-1/2}$	$F(\mathbf{z}) = 0.5\mathbf{z}'\Sigma^{-1}\mathbf{z}$ $\mathbf{f}(\mathbf{z}) = \Sigma^{-1/2}\mathbf{z}$ LEAST-SQUARES $\Sigma = I$: $\frac{1}{2}\lVert\boldsymbol{\theta}-\mathbf{z}\rVert_F^2$ MAHALANOBIS DIST: $0.5MD(\boldsymbol{\theta})$
MULTINOMIAL $N > 0$ NUMBER OF TRIALS $\mathbf{z} \in \mathbb{N}^n$, n EVENTS $\mathbf{z}_j$ FREQ. OF EVENT j	$p(\mathbf{z}\|\mathbf{q}) = \frac{N!}{\prod_{j=1}^n \mathbf{z}_j!} \prod_{j=1}^n \mathbf{q}_j^{\mathbf{z}_j}$ $\boldsymbol{\theta} = [\ln(\frac{\mathbf{q}_1}{\mathbf{q}_n}, \ldots, \frac{\mathbf{q}_{n-1}}{\mathbf{q}_n}),\ 0]$	$F(\boldsymbol{\theta}) = N\ln(\mathbf{1}'\exp(\boldsymbol{\theta}))$ $f(\boldsymbol{\theta}) = N\exp(\boldsymbol{\theta})/(\mathbf{1}'\exp(\boldsymbol{\theta}))$ $\mathbf{1}'\exp(\boldsymbol{\theta}) = 1 + \sum_{j=1}^{n-1}\exp(\boldsymbol{\theta}_i)$ SOFTMAX TRANSFER
POISSON $z \in \mathbb{N}$, z= # OF EVENTS THAT OCCURRED IN INTERVAL	$\theta \in \mathbb{N}$ AVERAGE # OF EVENTS $p(z\|\theta) = \theta^2\exp(-\theta)/z!$	UNNORMALIZED KL-DIVERGENCE $F(\theta) = \theta\ln(\theta) - \theta$ $\mathbf{f}(\theta) = \ln(\theta)$
WEIBULL $z = x^k$ FOR SOME $k > 0$ $\theta = -\lambda^k < 0$ λ SCALE, $1/\lambda$ RATE	$p(z\|\theta) = -\theta\exp(z\theta)p_0(z)$ $p_0(z) = x^{k-1}$ $k = 1$ IS EXPONENTIAL DIST. MODELS TIME TO FAILURE	$F(\theta) = -\ln(-\theta)$ $f(\theta) = -\theta^{-1}$

Table A.1: Detailed information about exponential family distributions and their associated transfers and Bregman divergences. Each distribution is a natural exponential family: $p_F(\mathbf{z}|\boldsymbol{\theta}) = \exp(\mathbf{z}'\boldsymbol{\theta} - F(\boldsymbol{\theta}))p_0(\mathbf{z})$. The minimization over the Bregman divergence can be simplified because terms only dependent on $\mathbf{z}$ can be dropped in the minimization: $\min_{\boldsymbol{\theta}} -\ln p_F(\mathbf{z}_i|\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} D_F(\boldsymbol{\theta}|f^{-1}(\mathbf{z})) = \min_{\boldsymbol{\theta}} F(\boldsymbol{\theta}) - F(f^{-1}(\mathbf{z})) - f(f^{-1}(\mathbf{z}))'(\boldsymbol{\theta}-\mathbf{z}) = \min_{\boldsymbol{\theta}} F(\boldsymbol{\theta}) - \mathbf{z}'\boldsymbol{\theta}$. The goal is to learn $f(\boldsymbol{\theta}) \approx \mathbf{z}$. For example, for data $\mathbf{x}$, the goal may be to learn $\hat{W}$ such that $f(\hat{W}\mathbf{x}) \approx f(W\mathbf{x}) = \mathbf{y}$. Note that for the tables in (Banerjee et al., 2005), $F = \psi$.

Appendix B

Generalized forward-reverse connection

B.1 Proof of Theorem 1

Theorem 1 [Forward-Reverse Equivalence] Given a $n \times T$ input matrix X and a $k \times T$ output matrix Y, such that $T > \text{rank}(X) = n$ and $T > \text{rank}(Y) = k$, there exist unique global minimizers

$$\begin{aligned} W^* &= \arg\min_W D_F(WX||\mathbf{f}^{-1}(Y)) \\ C^* &= \arg\min_C D_{F^\star}(CY||\mathbf{f}(X)) \\ \text{where} \quad & \mathbf{f}(W^*X)X' = YX' = Y\mathbf{f}^{-1}(C^*Y)'. \end{aligned}$$

Proof: Let F be a strictly convex function with $\text{Dom}(F) = \{WX : W \in \mathbb{R}^{k\times n}\}$, with any full rank X (i.e., X such that $W_1X \neq W_2X$ for $W_1 \neq W_2$). Then $G = F(\cdot X)$ has $\text{Dom}(G) = \mathbb{R}^{k\times n}$ (which is convex). For W_1, W_2 in $\text{Dom}(G)$ such that $W_1 \neq W_2$

$$\begin{aligned} G(\lambda W_1 + (1-\lambda)W_2) &= F((\lambda W_1 + (1-\lambda)W_2)X) = F(\lambda W_1X + (1-\lambda)W_2X) \\ &< \lambda F(W_1X) + (1-\lambda)F(W_2X) \quad \text{because } F \text{ is strictly convex and } W_1X \neq W_2X. \\ &= \lambda G(W_1) + (1-\lambda)G(W_2) \end{aligned}$$

Therefore, G is strictly convex. The optimization $\min_W G(W)$ therefore has a unique minimum. Notice that we can always linearize X, W and Y to make sure we are working with vectors.

For the relation, since W^* and C^* are global minimizers of $L(WX, Y)$ and $R(X, CY)$,

$$\begin{aligned} &\frac{d}{dW}L(W^*X, Y) = (\mathbf{f}(W^*X) - Y)\,X' = \mathbf{0} \quad (k \times n) \\ &\frac{d}{dC}R(X, C^*Y) = (\mathbf{f}^*(C^*Y) - X)\,Y' = \mathbf{0} \quad (n \times k) \\ \implies &(\mathbf{f}(W^*X) - Y)\,X' = ((\mathbf{f}^*(C^*Y) - X)\,Y')' \\ \implies &\mathbf{f}(W^*X)X' - YX' = Y\mathbf{f}^*(CY)' - YX' \\ \implies &\mathbf{f}(W^*X)X' = Y\mathbf{f}^*(C^*Y)' \end{aligned}$$

■

Although in general the relationship between W^* and C^* is implicit, in some cases an explicit conversion can be recovered.

Corollary 5. *If $X \in \mathbb{R}^{T \times T}$ has full column rank, $\operatorname{rank}(X) = T$, then*

$$W^* = \mathbf{f}^{-1}(Y\mathbf{f}^{-1}(C^*Y)'X'^{-1})X^{-1}.$$

Though this is a restrictive requirement, this corollary can be useful for identifying a closed-form mapping for when using kernels, since the kernel is often a square, full-rank matrix.

B.2 Kernelization, regularization, and instance weighting

As in the least-squares setting, the equivalence between forward and reverse prediction is preserved after adding kernels, regularizer and/or instance weighting, due to the relationship (3.11).

We start with regularization, since it is a key component of our regularized factor models Recall that the regularizer corresponds to a prior on the parameters, in this case, $p(W)$ and $p(C)$. One would expect that there is a strict relationship between the priors on W and C: intuitively a chosen prior on W should narrow the space of allowed priors on C. Instead, we find that, in fact, a unique W corresponds to a set of possible C which can have any prior.

To see why this is the case, first notice that

$$P(X, Y) = \int P(X, C|Y)P(Y)dC$$
$$P(X, Y) = \int P(Y, W|X)P(X)dW$$

As in Lemma 1, we can again represent this integral as an entropy regularized maximization,

$$\max_{q(\cdot)} \int q(W)P(Y, W|X)P(X)dW + H(q(\cdot)) = \max_{q(\cdot)} \int q(C)P(X, C|Y)P(Y)dC + H(q(\cdot)).$$

If we again drop the entropy regularizer, then we obtain the forward and reverse losses, because

$$\max_{q(\cdot)} \int q(W)P(Y, W|X)P(X)dW = \max_{C} P(Y, W|X)P(X) = \max_{C} P(Y|X, W)P(W|X)P(X)$$

Alternatively, if we could assume that $P(X, W|Y, C) = P(X|Y, C)$ and $P(Y, C|X, W) = P(Y|X, W)$, then we would obtain the following equalities.

$$P(X, W, Y, C) = P(X, W|Y, C)P(Y, C) = P(X|Y, C)P(Y, C)$$
$$P(X, W, Y, C) = P(Y, C|X, W)P(X, W) = P(Y|X, W)P(X, W)$$

Therefore, we obtain

$$P(X|Y, C)P(Y, C) = P(Y|X, W)P(X, W).$$

Now notice that $P(X,W) = P(W|X)P(X) = P(W)P(X)$ if the prior on W is not influenced by X. Similarly, $P(Y,C) = P(C)P(Y)$. As described in Section 2.1, the goal is to minimize the negative log likelihood. Since $-\log P(X|Y,C) = D_{F^\star}(CY||f(X))+$ constants and $-\log P(Y|X,W) = D_F(WX||f^{-1}(Y))+$ constants, we get

$$\min_W D_F(WX||f^{-1}(Y)) - \log p(W) \equiv \min_C D_{F^\star}(CY||f(X)) - \log p(C)$$

where $\log P(X)$ and $\log P(Y)$ are dropped since they do not affect the minimization.

Given the optimal W for a chosen forward prior, therefore, there exists a unique C for all possible priors that satisfies the reverse optimization. For example, assuming independent priors on C and Y, if one adds a convex, differentiable regularizer, $R : \mathbb{R}^{n\times k}$, to the forward training problem,

$$\arg\min_W D_F(WX||f^{-1}(Y)) + \alpha R(W) \tag{B.1}$$

then for any prior $p(C)$, we immediately obtain the solution equivalence

$$\mathbf{f}(W^*X)X' + \alpha\nabla R(W^*) = Y\mathbf{f}^{-1}(C^*Y)' - \nabla \log p(C^*) \tag{B.2}$$

using the same approach to get the equality in (3.11). The choice of regularizer/prior on W, therefore, does not restrict the choice of priors on C.

One way to obtain a unique equivalence between regularized forward and reverse prediction problems is to impose a constraint on the recovery of W from C and vice versa. A reasonable constraint on the equivalence is to choose the prior such that $D_F(XW^*||f^{-1}(Y)) = D_{F^\star}(YC^*||f(X))$, since we may want equal weight on the main loss and equal weight on the regularizers. In this setting, given W^*, we can obtain a unique equivalence using the optimization

$$\min_{C:\log p(C)=\log p(W^*)} D_{F^\star}(YC||f(X))$$

assuming that $-\log p(C)$ is a convex regularizer. In general, however, the relationship between these priors remains an important open problem. The selection of the priors for the forward and reverse problems is likely tied to the entropy regularizer in the formulation that maximizes over the priors $q(W)$ and $q(C)$, but requires further investigation.

Second, re-expressing the training problem in a *reproducing kernel Hilbert space* (RKHS) is an important extension: kernels increase modeling power and enable a large, even infinite number of features [1]. However, from the representer theorem (Kimeldorf and Wahba, 1971), the regularized loss (B.1) must admit an RKHS embedding provided the regularizer is of the form $R = \Omega(h(WW'))$ for a function h that is matrix nondecreasing (Argyriou et al., 2009).[2] This

[1] Note that one cannot simply incorporate a kernel into a non-linear transfer function: inner products, for example, are not invertible.

[2] A function h is matrix non-decreasing if $M \succeq N$ implies $h(M) \geq h(N)$ for $M \succeq 0$ and $N \succeq 0$.

form incorporates many useful regularizers, including the commonly used Frobenius norm, $\|W\|_F^2$, and trace norm (aka nuclear norm), $\|W\|_{tr}$. Thus, $W\mathbf{x}$ can be represented as $\sum_t \alpha_t k(\mathbf{x}_t, \mathbf{x})$ for some given values $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$. For a least squares regularizer, for example, since the form of $W = AX'$, the kernelized form of Equation (B.1) becomes

$$\arg\min_A D_F(AK||f^{-1}(Y)) + \alpha \ \mathrm{tr}(A'AK). \tag{B.3}$$

Using the corresponding reverse loss $D_{F^\star}(BY||\mathbf{f}(K))$ with regularizers $R(A) = -\log p(A)$ $G(B) = -\log p(B)$, we obtain the equivalence

$$\mathbf{f}(A^*K)K + \nabla R(A) = Y\mathbf{f}^{-1}(B^*Y)' + \nabla G(B). \tag{B.4}$$

A closed form recovery is simple for the case that $\nabla R(A) = 0$. Since K is invertible, we can use Corollary 5 to obtain the recovery $A = \mathbf{f}^{-1}((Y\mathbf{f}^{-1}(B^*Y)' + \nabla(B))K^{-1})K^{-1}$, or if $\nabla G(B) = 0$, $A = \mathbf{f}^{-1}(Y\mathbf{f}^{-1}(B^*Y)'K^{-1})K^{-1}$. This setting, $\nabla R(A) = 0$, arises for kernel PCA, for example [3].

Finally, instance weights can be incorporated while still preserving unique forward-reverse solution correspondences. The Fenchel dual of $F_\lambda(\mathbf{z}) = \lambda F(\mathbf{z})$ for weight $\lambda > 0$ is given by

$$F_\lambda^\star(\mathbf{y}) = \max_{\mathbf{z}} \mathbf{z}'\mathbf{y} - F_\lambda(\mathbf{z}) = \max_{\mathbf{z}} \mathbf{z}'\mathbf{y} - \lambda F(\mathbf{z}) = \lambda \max_{\mathbf{z}} \mathbf{z}'(\mathbf{y}/\lambda) - F(\mathbf{z}) = \lambda F^\star(\mathbf{y}/\lambda).$$

The corresponding gradients are $\mathbf{f}_\lambda(\mathbf{x}) = \lambda\mathbf{f}(\mathbf{x})$ and $\mathbf{f}_\lambda^\star(\mathbf{y}) = \mathbf{f}^{-1}(\mathbf{y}/\lambda)$. Despite the modified conjugates, the reverse problem simplifies to adding instance weights to the original reverse loss. To illustrate, consider an instance-weighted version of the forward loss minimization problem

$$\begin{aligned}
\min_W \sum_{t=1}^{t} \lambda_t D_F(WX_{:,t}||\mathbf{f}^{-1}(Y_t)) &= \min_W \sum_{t=1}^{T} \lambda_t \left[F(WX_{:,t}) - F(\mathbf{f}^{-1}(Y_{:,t})) - Y'_{:,t}(WX_{:,t} - \mathbf{f}^{-1}(Y_{:,t}))\right] \\
&= \min_W \sum_{t=1}^{T} \lambda_t \left[F(WX_{:,t}) - Y'_{:,t}WX_{:,t}\right] \\
&= \min_W \sum_{t=1}^{t} D_{F_{\lambda_t}}(WX_{:,t}||\mathbf{f}_{\lambda_t}^{-1}(\lambda_t Y_t)).
\end{aligned}$$

Using the above identity, we conclude that the corresponding reverse loss can be expressed

$$\begin{aligned}
&D_{F^*_{\lambda_t}}(\lambda_t CY_t||\mathbf{f}_{\lambda_t}(X_{:,t})) \\
&\quad = \lambda_t F^\star(\lambda_t CY_t\lambda_t^{-1}) - \lambda_t F^\star(\lambda_t\mathbf{f}(X_{:,t})\lambda_t^{-1}) - \mathbf{f}^{-1}(\lambda_t\mathbf{f}(X_{:,t})\lambda_t^{-1})(\lambda_t CY_t - \lambda_t\mathbf{f}(X_{:,t})) \\
&\quad = \lambda_t F^\star(CY_t) - \lambda_t F^\star(\mathbf{f}(X_{:,t})) - \lambda_t\mathbf{f}^{-1}(\mathbf{f}(X_{:,t}))(CY_t - \mathbf{f}(X_{:,t})).
\end{aligned}$$

[3] As described in the Appendix B.2.1, this can be derived differently for least-squares losses, with a different minimization, but results in the same solution

Since minimizing $D_{F^*_{\lambda_t}}$ is equivalent to minimizing the instance-weighted reverse loss $\lambda_t D_{F^\star}$,

$$\begin{aligned}\min_C \sum_{t=1}^{t} D_{F^*_{\lambda_t}}(\lambda_t CY_t || \mathbf{f}_\lambda(X_{:,t})) &= \min_C \sum_{t=1}^{t} \lambda_t \left[F^\star(CY_t) - X'_{:,t} CY_t\right] \\ &= \min_C \sum_{t=1}^{t} \lambda_t D_{F^\star}(CY_t || \mathbf{f}(X_{:,t}))\end{aligned}$$

the forward-reverse solution equivalence is retained for instance-weighting.

B.2.1 Simplifications for least-squares losses

The reverse losses given in Section 3.1.2 arise immediately by selecting the identity transfer in the previous section. For example, for an identity transfer, we obtain $A^*K(K+\alpha) = YY'(B^*)'$. We can, however, take advantage of some simplifications for least-squares losses to avoid some of the kernel inversions by considering a slightly modified reverse loss. The below are the losses considered in the first reverse prediction work (Xu et al., 2009).

For least-squares losses, the reverse solutions always have the form $C = XB$. For example, $B = Y^\dagger$ for $\min_C \operatorname{tr}\left((X - CY)'(X - CY)\right)$ which has solution $C = XY^\dagger$. Thus, a different kernelized training problem corresponding to (3.4) is given by

$$\min_B \ \operatorname{tr}\left((X - XBY)'(X - XBY)\right) = \min_B \ \operatorname{tr}\left((I - BY)'K(I - BY)\right) \tag{B.5}$$

where K corresponds to $X'X$ in some feature representation. It is easy to verify that the global minimizer is given by $B = Y^\dagger$. The forward solution can again be recovered from the reverse solution. Using the identity arising from the solutions of (3.3) and (B.5), $A(K+\alpha I) = Y = YY'B'$,

$$A = YY'B'(K + \alpha I)^{-1}.$$

Given the diagonal weighting matrix Λ, the weighted problem is

$$\min_B \ \operatorname{tr}\left(\Lambda(I - BY)'K(I - BY)\right)$$

where $B = \Lambda Y'(Y\Lambda Y')^{-1}$ is the global minimizer. Since $A(K\Lambda + \alpha I) = Y\Lambda = (Y\Lambda Y')'B'$, the forward solution can be recovered by

$$A = Y\Lambda Y'B'(K\Lambda + \alpha I)^{-1}.$$

Appendix C

Unsupervised learning algorithms

This appendix contains a list of unsupervised learning algorithms that are regularized factor models, in addition to independent components analysis which is closely related. These sections are mainly meant as a quick reference and summary of a large literature, often opting for citations to original proofs rather than recreating them here.

C.1 Linear representation learning

Linear representation learning techniques consist of using a Euclidean loss, either with or without kernels. These algorithms constitute many of the classical unsupervised learning algorithms. Algorithms using general Bregman divergences are discussed in the next section.

C.1.1 Probabilistic latent semantic indexing

Ding et al. (2006) showed that probabilistic latent semantic indexing is optimizing the same objective as non-negative matrix factorization.

C.1.2 Partial least squares

Partial Least Squares (PLS), also called *projection to latent structures*, maximizes the *covariance* between two spaces (Wold, 1985), as opposed to the correlation maximized in CCA. Orthonormal PLS is equivalent to CCA because correlation and covariance are equivalent for normal vectors (Sun et al., 2009b); see (Wegelin, 2000, Table 2) for a thorough comparison between PLS and CCA.

PLS attempts to find the eigenvectors of (De Bie et al., 2005)

$$\begin{bmatrix} 0 & XY' \\ YX' & 0 \end{bmatrix} \begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix} = \begin{bmatrix} C^{(1)} \\ C^{(2)} \end{bmatrix} \Lambda$$

Therefore, the PLS objective can be solved by simply taking the SVD of XY' or equivalently by obtaining the eigenvectors of $XY'YX'$. Partial least squares algorithms, however, rarely outright solve this system, but rather use an iterative deflation approach. Consequently, though the first eigenvector found is the same, solutions found by PLS are slightly different. For example, the PLS conjugate gradient optimization produces slightly different dictionaries that have empirically been shown to produce a closer fit (Phatak and de Hoog, 2002). This corresponds to a similar result that states that the columns of C produced by PLS form the basis of a Krylov space (Rosipal and Krämer, 2006), which is the objective of conjugate gradient algorithms.

C.1.3 Independent component analysis

The goal of independent component analysis (ICA) is to decompose the signal $\mathbf{x} \in \mathbb{R}^n$ into a linear combination of independent source signals, $\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n$:

$$\min_{C,\Phi} ||X - C\Phi|| \qquad \text{where row } \Phi_{i,:} \text{ is independent of row } \Phi_{j,:} \text{ for all } i, j.$$

Typically, to ensure that the signals are independent, it is assumed that $\Phi = WX$ and higher order statistics (such as the kurtosis or contrast criteria) of WX are minimized (Comon, 1994). For example, for $X = U\Sigma V'$, the kurtosis of WZ for $Z_{i,:} = U_{i,:}/(U_{i,:}\mathbf{1})^2$ is minimized: $\min_W (WZ)^4$. For Gaussian data, ICA and PCA are equivalent because kurtosis is zero: independence and uncorrelatedness are equivalent (Hyvärinen et al., 2009). By extending ICA to two data sets (Akaho et al., 1999; De Bie and De Moor, 2002), we get a similar extension on CCA which incorporates higher order statistics (as opposed to just correlation, which is a second order statistic).

Because the problem of enforcing independence of the rows of Φ can be encoded differently, there is not one unique optimization for ICA. The typical optimization is to find de-mixing matrix $W \in \mathbb{R}^{n \times n}$ such that $\Phi = WX$ such that WX has independent rows. Gordon (2003) showed that ICA is an instance of generalized^2 linear^2 models under a Bregman divergence with identity transfer and an additional transfer on the representation

$$\|C \operatorname{arctanh}(\Phi) - X\|_F^2 .$$

A flat minimum search algorithm, which consists of a multi-layer loss and regularizer that prefers low network complexity, was empirically shown to select independent source signals (Hochreiter and Schmidhuber, 1999). Another obvious option is to use kurtosis as a regularizer on Φ; optimizing such a regularizer, however, would be non-trivial.

C.2 Graph-based techniques

Many nonlinear manifold learning techniques can be viewed as kernel PCA,and so as a regularized factor models. The main difference is in their selection of an effective distance measure, i.e., kernel.

C.2.1 Isomap

Let $S_{i,j} = \|X_{:i} - X_{:j}\|$ be the matrix of squared distances between points and $\mathbf{e} = T^{-1/2}[1, \ldots, 1]'$ is the uniform vector of unit length. Ham et al. (2004) showed that Isomap is kernel PCA, with the kernel $K = -\frac{1}{2}(I - \mathbf{e}\mathbf{e}')S(I - \mathbf{e}\mathbf{e}')$. Therefore, the objective of Isomap is to minimize $L\left(C\Phi\ ;\ X\right) = \|K - C\Phi\|_F^2$ and Isomap is one algorithm to minimize this loss.

C.2.2 Laplacian eigenmaps

Let $L \in \mathbb{R}^{T\times T}$ be the generalized graph Laplacian, which depends on the chosen adjacency matrix, i.e., either having indicators for adjacency

$$L_{i,j} = \begin{cases} -1 & \text{if } i \text{ is a neighbour of } j \\ \sum_k \mathbb{1}(k \text{ is a neighbour of } i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

or Gaussian drop-off

$$L_{i,j} = \begin{cases} -\exp(-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2 / (2\sigma^2)) & \text{if } i \text{ is a neighbour of } j \\ \sum_{k \text{ is a neighbour of } i} \exp(-\left\|\mathbf{x}_i - \mathbf{x}_k\right\|_2 / (2\sigma^2)) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

for some parameter $\sigma > 0$. Ham et al. (2004) showed that Laplacian eigenmaps is kernel PCA, with the kernel $K = L^{\dagger}$. Therefore, the objective of Laplacian eigenmaps is to minimize $L\left(C\Phi\ ;\ X\right) = \|K - C\Phi\|_F^2$ and Laplacian eigenmaps can be considered as one algorithm to minimize this loss. The choice of this kernel can be viewed as visitation frequencies or commute times, i.e., K_{ij} is the expected time spent in node j starting from node i (Ham et al., 2004).

Kernel locality preserving projections has also been shown to be equivalent to Laplacian eigenmaps (He and Niyogi, 2004). In addition, generally if the span of X is invariant under the normalized Laplacian matrix, then locality preserving projections is equivalent to Laplacian eigenmaps (Kokiopoulou et al., 2011, Proposition 6.4). Finally, Brand (2003) has also illustrated that several nonlinear dimensionality reduction techniques can be viewed as using different choices of kernel, incorporating additional algorithms like charting and automatic alignment.

C.2.3 Locally linear embeddings

Let $W \in \mathbb{R}^{T \times p}$ be the weight matrix where the ith row contains linear coefficient to optimally reconstruct $\mathbf{x}_i$ from its p nearest neighbours and $W\mathbf{1} = \mathbf{1}$. Let $M = (I - W')(I - W)$. Then, for $\mathbf{e} = T^{-1/2}[1, \ldots, 1]'$ the uniform vector of unit length and $\lambda_{\max}(M)$ the maximum eigenvalue of M, locally linear embeddings (LLE) is kernel PCA, with the kernel $K = (I - \mathbf{e}\mathbf{e}')(\lambda_{\max}(M)I - M)(I - \mathbf{e}\mathbf{e}')$ (Ham et al., 2004).

In certain situations, neighbourhood preserving projections is equivalent to LLE; see (Kokiopoulou et al., 2011, Proposition 6.4). As with locality preserving projections, kernel neighbourhood preserving projections is euivalent to LLE (Kokiopoulou et al., 2011, Proposition 7.3).

C.2.4 Metric multi-dimensional scaling

Metric multi-dimensional scaling has been shown to be equivalent to kernel PCA (Williams, 2002), when the kernel function $k(\mathbf{x}, \mathbf{y})$ is isotropic, meaning it depends only on $\|\mathbf{x} - \mathbf{y}\|$. Therefore, kernelized unregularized factor models with an isotropic kernel, $K(\mathbf{x}, \mathbf{y}) = g(\|\mathbf{x} - \mathbf{y}\|)$ for some function g, and loss $L(C\Phi\ ;\ X) = \|K - C\Phi\|_F^2$ is equivalent to metric multi-dimensional scaling.

C.2.5 Ratio cut

Ratio cut is similar to normalized cut, except that it does not normalize the data with $\Lambda = \operatorname{diag}(\mathbf{1}K)$. It is therefore equivalent to a standard eigenvalue problem (rather than a generalized one), where the solution is k eigenvectors corresponding to the minimum eigenvalues of the graph Laplacian, $L = \Lambda - K$. Correspondingly, the solution corresponds to the top k eigenvectors of the pseudo-inverse matrix, $L^\dagger$, meaning it is kernel PCA with $K = L^\dagger$ (Kokiopoulou et al., 2011, Proposition 7.4). Therefore, ratio cut is equivalent to Laplacian eigenmaps.

C.2.6 Projection algorithms

There are several algorithms that can be viewed as a projections of the previous algorithms. Locality preserving projections is a projected version of Laplacian eigenmaps; it computes the minimum eigenvalue of the generalized eigenvalue problem $XLX'\mathbf{v} = \lambda XDX'\mathbf{v}$ (Kokiopoulou et al., 2011). Neighbourhood preserving projections is a projected version of LLE; it computes the minimum eigenvalue of the generalized eigenvalue problem $XLX'\mathbf{v} = \lambda XDX'\mathbf{v}$ (Kokiopoulou et al., 2011). Manifold alignment without correspondence (Wang and Mahadevan, 2009) is equivalent to locality preserving projections, with $Z = \left[\begin{smallmatrix} X & 0 \\ 0 & Y \end{smallmatrix}\right]$.

These algorithms can be written as regularized factor models, simply using the trace norm loss and including $XDX' = I$ as a constraint. This view, however, does not gain further insight into

representing the problems with known distributional properties, i.e., with Bregman divergences. The connection would be obtained if the generalized eigenvalue problem could be written as a least-squares optimization, for more general cases than in Corollary 1.

C.3 Linear clustering

Clustering can also be thought of as a factorization, where the representation Φ corresponds to the unknown clusters and the basis C corresponds to the means. Each cluster can be considered an explanatory factor, with different constraints on the factors and on their weights, C. In classification, recall that the rows in the target label matrix, Y, indicate the class label of the corresponding instance. If the target labels are missing, we would like to guess a label matrix Φ that satisfies the same constraints, namely that $\Phi \in \{0,1\}^{t\times k}$ and $\Phi\mathbf{1} = \mathbf{1}$.

Proposition 10. ***K-means clustering*** *solves a constrained factorization with a least-squares loss and* $\mathcal{F} = \{\Phi \in \{0,1\}^{k\times T},\ \mathbf{1}\Phi = \mathbf{1}\}$,

$$\min_{\Phi:\Phi\in\{0,1\}^{k\times T},\,\mathbf{1}\Phi=\mathbf{1}} \min_{C} \ tr\left((X - C\Phi)'(X - C\Phi)\right) \tag{C.1}$$

Proof: The closed form solution for C given Φ is $C = X\Phi^{\dagger}$, giving

$$(\text{C.1}) = \min_{\Phi:\Phi\in\{0,1\}^{k\times T},\,\mathbf{1}\Phi=\mathbf{1}} \operatorname{tr}\left((I - \Phi^{\dagger}\Phi)'X'X(I - \Phi^{\dagger}\Phi)\right)$$

This loss is a sum of squares of the difference $X(I - \Phi^{\dagger}\Phi) = X - X\Phi'(\Phi\Phi')^{-1}\Phi$. To interpret this difference matrix: [1]

1. $X\Phi'$ is a $n \times k$ matrix where column i is the sum of rows in X that have class i in Φ:

$$(X\Phi')_{:,i} = \sum_{j\in\{1,\ldots,T\},\ \text{s.t.}\Phi_{i,j}=1} X_{:,j}$$

2. The diagonal matrix $\Phi\Phi'$ contains the count of ones in each row of Φ; therefore $(\Phi\Phi')^{-1}$ is a diagonal matrix of reciprocals of row counts.

3. Combining facts 1 and 2 shows that $C = X\Phi'(\Phi\Phi')^{-1}$ is a $n \times k$ matrix whose column i is the *mean* of the columns in X that correspond to class i in Φ.

4. Finally, $C\Phi = X\Phi'(\Phi\Phi')^{-1}\Phi$ is a $n \times T$ matrix where column i contains the mean corresponding to class i in Φ.

[1] These observations draw on some insights from Peng and Wei (2007).

Therefore, $X - X\Phi'(\Phi\Phi')^{-1}\Phi$ is a $n \times T$ matrix of columns from X with the mean column for each corresponding class subtracted. The problem (C.1) can now be seen to be equivalent to assigning k centers, encoded by $C = X\Phi'(\Phi\Phi')^{-1}$, and assigning each column in X to a center, encoded by Φ, so as to minimize the sum of the squared distances between each column and its assigned center. ■

The intuition behind this view of k-means clustering is that the representation, Φ, is the cluster index and the dictionary C is the means of those clusters. The new representation for X, therefore, can be seen as the mean of its cluster or even its cluster index, depending on if Φ or $C\Phi$ is used as the new representation.

Proposition 11. ***Kernel k-means clustering*** *solves a constrained factorization with a kernelized least-squares loss and* $\mathcal{F} = \{\Phi \in \{0,1\}^{k \times T},\ \mathbf{1}\Phi = \mathbf{1}\}$,

$$\min_{\Phi:\Phi\in\{0,1\}^{k\times T},\,\mathbf{1}\Phi=\mathbf{1}} \min_{C}\ tr\left((K - C\Phi)'(K - C\Phi)\right)$$

C.4 Generalized clustering

C.4.1 Exponential family k-means clustering

Banerjee et al. (2005) generalized the centroid-based hard clustering problem to clustering using any Bregman divergence. In Proposition 12, we show that Bregman clustering solves a regularized factorization, despite the fact that Banerjee et al. (2005) reach this generalization using the idea of Bregman information rather than reverse prediction or regularized factor models.

Proposition 12. ***Clustering with Bregman divergences*** *solves a constrained factorization with* $\mathcal{F} = \{\Phi \in \{0,1\}^{k \times T},\ \mathbf{1}\Phi = \mathbf{1}\}$, *any Bregman divergence and non-transfered* $C\Phi$*:*

$$\min_{\Phi:\Phi\in\{0,1\}^{k\times T},\,\mathbf{1}\Phi=\mathbf{1}} \min_{C}\ D_{F^*}(\mathbf{f}(C\Phi)||\mathbf{f}(X)) \tag{C.2}$$

Proof: For $\mathcal{F} = \{\Phi \in \{0,1\}^{k \times T},\ \mathbf{1}\Phi = \mathbf{1}\}$, k-means clustering with Bregman divergences was framed under the minimization $\min_{\Phi\in\mathcal{F}} \min_C D_F(X||C\Phi)$ (Banerjee et al., 2005). Thus, we can apply the same argument as in Proposition 4. ■

This formulation of Bregman clustering, however, only permits *linear* transfers: $X \approx C\Phi$, learned under a Bregman divergence. Using insights from reverse prediction, the clear extension that incorporates non-linear transfers is the optimization

$$\min_{\Phi:\Phi\in\{0,1\}^{k\times T},\,\mathbf{1}\Phi=\mathbf{1}} \min_{C}\ D_{F^*}(C\Phi||\mathbf{f}(X)). \tag{C.3}$$

An algorithm for this modification on Bregman clustering is given in Chapter 6, Algorithm 8, with derivation in Appendix F.1.

Banerjee et al. (2005) also proposed an algorithm for Bregman mixture models, i.e., soft clustering where the constraint $\Phi \in \{0,1\}^{k\times T}$ is relaxed to $\Phi \in [0,1]^{k\times T}$. This optimization, however, and corresponding simplification do not correspond exactly to a factorization. Rather, the expectation-maximization objective is tackled, which is related to the hard clustering regularized factor model as a smoothness parameter $\rho \to \infty$. Roweis and Ghahramani (1999) also showed that mixture model clustering and factor analysis are related. Based on the discussion in Section 8.1.1, to incorporate mixture model clustering into regularized factor models, likely a regularizer on Φ will need to be considered. In Appendix F.1, we illustrate that the expectation-maximization objective provides an upper bound on our reverse prediction objective, and use that to derive a a non-linear Bregman mixture model algorithm similar to that of Banerjee et al. (2005). This algorithm is given in Chapter 6, Algorithm 9.

C.4.2 Generalized normalized cut

As with exponential family PCA, one can add regularization, kernels and instance weights to Bregman divergence clustering. For example, by adding instance weights $\Lambda = \operatorname{diag}(K\mathbf{1})$, we obtain a novel algorithm that generalizes normalized cut to Bregman divergences. With an identity transfer function (producing a least-squares matching loss), this algorithms reduces to normalized cut.

Proposition 13. ***Bregman normalized cut*** *solves a constrained factorization with* $\mathcal{F} = \{\Phi \in \{0,1\}^{k\times T}, \mathbf{1}\Phi = \mathbf{1}\}$ *and* $\Lambda = \operatorname{diag}(K\mathbf{1})$ *instance-weighted Bregman divergence for positive definite* K

$$\min_{\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \min_{C} D_{F_\Lambda}(\Lambda X \,||\, \mathbf{f}^{-1}(C\Phi)) = \min_{\Phi\in\{0,1\}^{k\times T},\, \mathbf{1}\Phi=\mathbf{1}} \min_{C} \sum_{t=1}^{T} \lambda_t D_{F^*}(C\Phi_{:,t} \,||\, \mathbf{f}(\lambda_t X_{:,t})).$$

C.4.3 Linde-Buzo-Gray algorithm

Banerjee et al. (2005) showed that the Linde-Buzo-Gray algorithm, used in speech coding, is an instance of Bregman clustering with Bregman divergence equal to the Itakura-Saito distance. This distance is often chosen for speech data because speech power spectra follow exponential densities $p(\mathbf{x}) = \lambda \exp(-\lambda \mathbf{x})$. Therefore, we get that **Linde-Buzo-Gray algorithm** is a constrained regularized factorization with non-transfered $C\Phi$ and $F(\mathbf{x}) = -\log(\mathbf{x})$ giving an exponential distribution on the data $p(X) = \lambda \exp(-\lambda x)$ and Bregman divergence equal to the Itakura-Saito distance.

C.4.4 Information theoretic clustering and Information bottleneck

Banerjee et al. (2005) showed that information-theoretic clustering is a subset of Bregman hard clustering algorithms, with Bregman divergence equal to KL-divergence. If the hard clustering is

relaxed to soft clustering, then the optimization is equal to the expectation-maximization algorithm on a mixture of multinomials. This soft clustering formulation corresponds to the information bottleneck. Banerjee et al. (2005, Theorem 7 and 8) also showed that a large class of rate distortion problems could be represented as a Bregman soft clustering problem, with a β-weighted Bregman divergence, or equivalently with potential function βF for the trade-off parameter β in rate distortion problems.

Appendix D

Convex factorization appendix

D.1 Preliminaries

The following known results are useful for the proofs in the remainder of Appendix D.

Lemma 5. *For* $Z \in \mathbb{R}^{n\times m}$, $\|Z\|_{\diamond,1}^* = \|Z\|_{\diamond^*,\infty} = \max_{i\in\{1,\dots,n\}}(\|Z\|_\diamond^*)$.

Proof: This conjugate norm follows from (Bradley and Bagnell, 2009, Lemma 3), where we can stack Z into a column vector, set the indices $s_i \in I$ to correspond to the rows i, the norm $\|\cdot\|_{s_i} = \|\cdot\|_\diamond$ and the outer norm to correspond to $\|\cdot\|_1$. ■

Throughout the proofs, we commonly use the fact that most convex optimization problems satisfy strong duality. As discussed in Section A.2.1, if Slater's condition, strong duality is guaranteed. The following known theorem gives another setting where strong duality holds.

Theorem 11 ((Rockafellar, 1970, Cor. 37.3.2))**.** *For any function* $L : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$*, if* $L(\cdot, \mathbf{y})$ *is convex for any* $\mathbf{y} \in \mathcal{Y}$*,* $L(\mathbf{x}, \cdot)$ *is concave for any* $\mathbf{x} \in \mathcal{X}$*,* $\mathcal{Y}$ *is finite-dimensional, convex and closed,* $\mathcal{X}$ *is convex and one of* $\mathcal{X}$ *or* $\mathcal{Y}$ *is compact, the strong duality holds, i.e.,*

$$\inf_{\mathbf{x}\in\mathcal{X}} \sup_{\mathbf{y}\in\mathcal{Y}} L(\mathbf{x},\mathbf{y}) = \sup_{\mathbf{y}\in\mathcal{Y}} \inf_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x},\mathbf{y})$$

Orthonormal bases and the singular value decomposition: The row-space, column-space and null-space of a matrix $A \in \mathbb{R}^{m\times n}$ can be characterized by the singular value decomposition. Recall that the row-space of A is the set of all possible linear combinations of the rows of A; the column-space of A is the set of all possible linear combinations of the columns of A; and the null-space of A is the set of all orthogonal vectors (the orthogonal complement to the row-space of A).

$$\begin{aligned}
\text{row}(A) &= \{\mathbf{v} \in \mathbb{R}^n : \text{there exists } \mathbf{u} \text{ such that } \mathbf{v} = A^\top \mathbf{u}\} \\
\text{col}(A) &= \{\mathbf{u} \in \mathbb{R}^m : \text{there exists } \mathbf{v} \text{ such that } \mathbf{u} = A\mathbf{v}\} \\
\text{null}(A) &= \{\mathbf{v} \in \mathbb{R}^n : A\mathbf{v} = \mathbf{0}\}.
\end{aligned}$$

The matrix A can be seen as a linear transformation taking $\mathbf{v}$ from the row-space of A to a vector $\mathbf{u} = A\mathbf{v}$ in the column-space of A.

The SVD relates the orthogonal basis of the row-space of A with the orthogonal basis of the column-space of A. For $k = \max(m, n)$ and $r = \operatorname{rank}(A) \leq \min(m, n)$, the svd is $A = U\Sigma V'$, with $\Sigma \in \mathbb{R}^{m \times n}$, $\Sigma_{ii} > 0$ for $i \leq r$, zero elsewhere; orthonormal $U = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$ $\mathbf{u}_i \in \mathbb{R}^m$; and orthonormal $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$, for $\mathbf{v}_i \in \mathbb{R}^n$. This gives $AV = U\Sigma VV' = U\Sigma$. We can therefore characterize the orthonormal basis of $\operatorname{col}(A)$ as $[\mathbf{u}_1, \ldots, \mathbf{u}_r]$, the orthonormal basis of $\operatorname{row}(A)$ as $[\mathbf{v}_1, \ldots, \mathbf{v}_r]$ and the orthonormal basis of $\operatorname{null}(A)$ as $[\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n]$.

D.2 Proof of Theorem 2

Theorem 2 *For any vector norm $\|\cdot\|_\diamond$ with conjugate $\|\cdot\|_{\diamond^*}$, and any bounded closed set $\mathcal{C} \subset \mathbb{R}^n$ such that $\operatorname{span}(\mathcal{C}) = \mathbb{R}^n$ and $C \in \mathcal{C}^\infty$ indicating that C has an unrestricted number of columns,*

$$\min_{C \in \mathcal{C}^\infty} \min_{\Phi} L(C\Phi; X) + \alpha \|\Phi\|_{\diamond,1} \tag{4.1}$$

$$= \min_{Z} L(Z; X) + \alpha \|Z'\|^*_{(\mathcal{C},\diamond^*)} \tag{4.2}$$

with $\|Z'\|^*_{(\mathcal{C},\diamond^*)} = \max_{\|Y\|_{(\mathcal{C},\diamond^*)} \leq 1} \operatorname{tr}(Y'Z')$ for induced norm $\|Y\|_{(\mathcal{C},\diamond^*)} = \max_{\mathbf{z} \in \mathcal{C}} \|Y\mathbf{z}\|_{\diamond^*}$ defined in Lemma 3.

Proof:

$$\begin{aligned}(4.1) &= \min_{Z} \min_{C \in \mathcal{C}^\infty} \min_{\Phi : C\Phi = Z} L(Z; X) + \alpha \|\Phi\|_{\diamond,1} \\ &= \min_{Z} L(Z; X) + \alpha \min_{C \in \mathcal{C}^\infty} \min_{\Phi : C\Phi = Z} \|\Phi\|_{\diamond,1}. \end{aligned} \tag{D.1}$$

Consider the inner minimization in (D.1). Fix any Z, $k \in \mathbb{N}$ and $C \in \mathcal{C}^k$, and introduce Lagrange multiplier Γ for constraint $C\Phi = Z$

$$\min_{\Phi : C\Phi = Z} \|\Phi\|_{\diamond,1} = \min_{\Phi} \max_{\Gamma} \|\Phi\|_{\diamond,1} + \operatorname{tr}(\Gamma'(Z - C\Phi)). \tag{D.2}$$

If C does not span the columns of Z, then the constraint $C\Phi = Z$ is infeasible, and (D.2) is unbounded; hence such a C cannot participate in a minimum of (D.1) and so any C selected in (D.1) must span the columns of Z. Given such a C, a feasible Φ exists, meaning Slater's condition is satisfied and strong Lagrange duality holds (Boyd and Vandenberghe, 2004, §5.2.3). We can therefore swap the min and max:

$$(D.2) \quad = \quad \max_{\Gamma} \min_{\Phi} \|\Phi\|_{p,1} + \operatorname{tr}(\Gamma'(Z - C\Phi)). \tag{D.3}$$

Since the dual norm of $\|\cdot\|_{\diamond,1}$ is $\|\cdot\|_{\diamond^*,\infty}$ by Lemma 5, by norm duality:

$$
\begin{aligned}
(D.3) &= \max_{\Gamma}\min_{\Phi}\max_{\|V\|_{\diamond^*,\infty}\le 1}\operatorname{tr}(V'\Phi)+\operatorname{tr}(\Gamma'(Z-C\Phi)) && \triangleright \text{strong duality holds so swap min and max}\\
&= \max_{\Gamma}\max_{\|V\|_{\diamond^*,\infty}\le 1}\min_{\Phi}\operatorname{tr}(\Gamma'Z)+\operatorname{tr}(\Phi'(V-C'\Gamma)) && \triangleright \text{because } \operatorname{tr}(AB)=\operatorname{tr}(BA)=\operatorname{tr}(A'B')\\
&= \max_{\|V\|_{\diamond^*,\infty}\le 1}\ \max_{\Gamma: C'\Gamma = V}\operatorname{tr}(\Gamma'Z) && \triangleright \text{by eliminating } \Phi\\
&= \max_{\Gamma:\|C'\Gamma\|_{\diamond^*,\infty}\le 1}\operatorname{tr}(\Gamma'Z) && \triangleright \text{by substituting } V = C'\Gamma.
\end{aligned}
$$

Therefore, we have established

$$
\begin{aligned}
(D.1) &= \min_{Z} L(Z;X)+\alpha\min_{C\in\mathcal{C}^\infty}\max_{\Gamma:\|C'\Gamma\|_{\diamond^*,\infty}\le 1}\operatorname{tr}(\Gamma'Z) && \triangleright \text{by simplifying (D.2)}\\
&= \min_{Z} L(Z;X)+\alpha\max_{\Gamma:\|\Gamma'\|_{(\mathcal{C},\diamond^*)}\le 1}\operatorname{tr}(\Gamma'Z) && \triangleright \text{by definition of } \|\cdot\|_{(\mathcal{C},\diamond^*)} \text{ in Lemma 3}\\
&= \min_{Z}\ L(Z;X)+\alpha\|Z'\|^*_{(\mathcal{C},\diamond^*)} && \triangleright \text{by norm duality.}
\end{aligned}
$$

■

D.3 Proof of Theorem 7

The proof follows from the two lemmas, 6 and 7.

Lemma 6. *For* $\|\Phi\|_{2,1}$, $\mathcal{C}_{2,2}=\{\mathbf{c}=\begin{bmatrix}\mathbf{c}^{(1)}\\ \mathbf{c}^{(2)}\end{bmatrix} : \|\mathbf{c}^{(1)}\|_2\le\beta_1, \|\mathbf{c}^{(2)}\|_2\le\beta_2\}\subset\mathbb{R}^{n_1+n_2}$,

$$
\|Z\|^*_{(\mathcal{C}_{2,2},2)} = \max_{0\le\eta\le 1}\|E_\eta^{-1}Z\|_{\mathrm{tr}} \qquad \textit{where } E_\eta := \begin{bmatrix}\beta_1/\sqrt{\eta}\,I_{n-1} & 0\\ 0 & \beta_2/\sqrt{1-\eta}\,I_{n_2}\end{bmatrix}.
$$

Proof: As a preliminary, note that $\mathcal{C}_{2,2}$ is bounded and closed set with $\operatorname{span}(\mathcal{C}_{2,2})=\mathbb{R}^{n_1+n_2}$ for $\mathbf{c}^{(1)}\in\mathbb{R}^{n_1}$ and $\mathbf{c}^{(2)}\in\mathbb{R}^{n_2}$, and so satisfies the conditions of Theorem 2. Therefore, $\|Z'\|^*_{(\mathcal{C}_{2,2},2)} = \min_{C\in\mathcal{C}^\infty_{2,2}}\min_{\Phi:C\Phi=Z}\|\Phi\|_{\diamond,1}$ defines a norm on Z.

Step 1: Characterize the dual norm as

$$
\|\Gamma\|_{(\mathcal{C}_{2,2},2)} = \min_{\rho\ge 0}\|D_\rho\Gamma\|_{\mathrm{sp}} \qquad \text{where } D_\rho = \begin{bmatrix}\sqrt{\beta_1^2+\beta_2^2\rho}\,I_{n_1} & 0\\ 0 & \sqrt{\beta_2^2+\beta_1^2/\rho}\,I_{n_2}\end{bmatrix}. \tag{D.4}
$$

We will use two diagonal matrices, $I_1=\operatorname{diag}([\mathbf{1}_{n_1};\mathbf{0}_{n_2}])$ and $I_2=\operatorname{diag}([\mathbf{0}_{n_1};\mathbf{1}_{n_2}])$. Starting with the definition,

$$
\|\Gamma\|_{(\mathcal{C}_{2,2},2)} = \max_{\{\mathbf{c}:\|\mathbf{c}^{(1)}\|_2=\beta_1,\,\|\mathbf{c}^{(2)}\|_2=\beta_2\}}\|\mathbf{c}'\Gamma\|_2
$$

we get

$$
\begin{aligned}
\|\Gamma\|^2_{(\mathcal{C}_{2,2},2)} &= \max_{\{\mathbf{c}:\|\mathbf{c}^{(1)}\|_2=\beta_1,\,\|\mathbf{c}^{(2)}\|_2=\beta_2\}}\mathbf{c}'\Gamma\Gamma'\mathbf{c} && \triangleright \text{replace } \Phi=\mathbf{c}\mathbf{c}' \qquad (D.5)\\
&= \max_{\{\Phi:\Phi\succeq 0,\operatorname{rank}(\Phi)=1\ \operatorname{tr}(\Phi I_1)\le\beta_1^2,\,\operatorname{tr}(\Phi I_2)\le\beta_2^2\}}\operatorname{tr}(\Phi\Gamma\Gamma') && \triangleright \text{drop the rank constraint}\\
&= \max_{\{\Phi:\Phi\succeq 0,\,\operatorname{tr}(\Phi I_1)\le\beta_1^2,\,\operatorname{tr}(\Phi I_2)\le\beta_2^2\}}\operatorname{tr}(\Phi\Gamma\Gamma').
\end{aligned}
$$

We can drop the rank constraint using the fact that when maximizing a convex function, one of the extreme points in the constraint set $\{\Phi : \Phi \succeq 0, \operatorname{tr}(\Phi I_1) \leq \beta_1^2, \operatorname{tr}(\Phi I_2) \leq \beta_2^2\}$ must be optimal and is known to have rank at most one in this case (Pataki, 1998).

Next, form the Lagrangian

$$L(\Phi; \lambda_1, \lambda_2, \Lambda) = \operatorname{tr}(\Phi\Gamma\Gamma') + \operatorname{tr}(\Phi\Lambda) + \lambda_1(\beta_1^2 - \operatorname{tr}(\Phi I_1)) + \lambda_2(\beta_2^2 - \operatorname{tr}(\Phi I_2))$$

where $\lambda_1 \geq 0$, $\lambda_2 \geq 0$ and $\Lambda \succeq 0$. The primal variable Φ can be eliminated by formulating the equilibrium condition $\partial L/\partial\Phi = \Gamma\Gamma' + \Lambda - \lambda_1 I_1 - \lambda_2 I_2 = 0$, which implies $\Gamma\Gamma' - \lambda_1 I_1 - \lambda_2 I_2 \preceq 0$. Therefore, we achieve the equivalent dual formulation

$$(\text{D.5}) = \min_{\{\lambda_1 \geq 0,\, \lambda_2 \geq 0,\, \Lambda \succeq 0\}} \max_{\Phi} L(\Phi; \lambda_1, \lambda_2, \Lambda) = \min_{\{\lambda_1 \geq 0,\, \lambda_2 \geq 0,\, \lambda_1 I_1 + \lambda_2 I_2 \succeq \Gamma\Gamma'\}} \beta_1^2\lambda_1 + \beta_2^2\lambda_2. \quad \text{(D.6)}$$

Now observe that for $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$, the relation $\Gamma\Gamma' \preceq \lambda_1 I_1 + \lambda_2 I_2$ holds if and only if $D_{\lambda_2/\lambda_1}\Gamma\Gamma' D_{\lambda_2/\lambda_1} \preceq D_{\lambda_2/\lambda_1}(\lambda_1 I_1 + \lambda_2 I_2) D_{\lambda_2/\lambda_1} = (\beta_1^2\lambda_1 + \beta_2^2\lambda_2) I$, i.e., iff the maximum eigenvalues of $D_{\lambda_2/\lambda_1}\Gamma$ are less than or equal to $\beta_1^2\lambda_1 + \beta_2^2\lambda_2$. Therefore,

$$(D.6) = \min_{\{\lambda_1 \geq 0,\, \lambda_2 \geq 0,\, \|D_{\lambda_2/\lambda_1}\Gamma\|_{sp}^2 \leq \beta_1^2\lambda_1 + \beta_2^2\lambda_2\}} \beta_1^2\lambda_1 + \beta_2^2\lambda_2 \quad \text{(D.7)}$$

The third constraint must be met with equality at the optimum due to continuity; otherwise, we would be able to further decrease the objective, which contradicts optimality. A standard compactness argument would establish the existence of minimizers. We finally obtain

$$(\text{D.7}) = \min_{\lambda_1 \geq 0, \lambda_2 \geq 0} \|D_{\lambda_2/\lambda_1}\Gamma\|_{\text{sp}}^2 = \min_{\rho \geq 0} \|D_\rho \Gamma\|_{\text{sp}}^2.$$

Step 2 Show that $\|Z'\|_{(\mathcal{C}_{2,2},2)}^* = \max_{\rho \geq 0} \|D_\rho^{-1} Z\|_{\text{tr}}$.

$$\begin{aligned}
\|Z\|_{(\mathcal{C}_{2,2},2)}^* &= \max_{\Gamma : \|\Gamma\|_{(\mathcal{C}_{2,2},2)} \leq 1} \operatorname{tr}(\Gamma' Z) && \triangleright \text{ by definition} \\
&= \max_{\rho \geq 0} \max_{\Gamma : \|D_\rho \Gamma\|_{\text{sp}} \leq 1} \operatorname{tr}(\Gamma' Z) && \triangleright \text{ using (D.4): } \|\Gamma\|_{(\mathcal{C}_{2,2},2)} = \min_{\rho \geq 0} \|D_\rho \Gamma\|_{\text{sp}} \\
&= \max_{\rho \geq 0} \max_{\tilde{\Gamma} : \|\tilde{\Gamma}\|_{\text{sp}} \leq 1} \operatorname{tr}(\tilde{\Gamma}' D_\rho^{-1} Z) && \triangleright \text{ change of variables } \tilde{\Gamma} = D_\rho \Gamma \\
&= \max_{\rho \geq 0} \|D_\rho^{-1} Z\|_{\text{tr}} && \triangleright \text{ because trace norm is conjugate of spectral norm.}
\end{aligned}$$

Step 3 Re-parametrize with $\rho = \frac{\beta_1^2(1-\eta)}{\beta_2^2 \eta}$, with $0 \leq \eta \leq 1$ for $\rho \geq 0$.

Because $\frac{\beta_1^2}{\beta_2^2\rho + \beta_1^2} = \frac{\beta_1^2}{\beta_2^2 \frac{\beta_1^2(1-\eta)}{\beta_2^2\eta} + \beta_1^2} = \frac{\beta_1^2}{\beta_1^2(\frac{(1-\eta)}{\eta} + 1)} = \frac{\beta_1^2}{\beta_1^2(\frac{(1-\eta)}{\eta} + 1)} = \eta$, we get

$$E_\eta = D_{\frac{\beta_1^2(1-\eta)}{\beta_2^2\eta}} \text{ and } D_\rho = E_{\frac{\beta_1^2}{\beta_2^2\rho + \beta_1^2}} \qquad \text{and so } \max_{\rho \geq 0} \|D_\rho^{-1} Z\|_{\text{tr}} = \max_{0 \leq \eta \leq 1} \|E_\eta^{-1} Z\|_{\text{tr}}.$$

■

Lemma 7. *$h(\eta) := \|E_\eta^{-1} Z\|_{\mathrm{tr}}$ is concave in η over $[0,1]$.*

Proof: Expand $h(\eta)$ into

$$h(\eta) = \left\| \begin{bmatrix} \sqrt{\frac{\eta}{\beta_1^2}} Z^{(1)} \\ \sqrt{\frac{1-\eta}{\beta_2^2}} Z^{(2)} \end{bmatrix} \right\|_{\mathrm{tr}} = \mathrm{tr}\left(\sqrt{\frac{\eta}{\beta_1^2} Z^{(1)\prime} Z^{(1)} + \frac{1-\eta}{\beta_2^2} Z^{(2)\prime} Z^{(2)}} \right)$$

where $\mathrm{tr}(\sqrt{\cdot})$ means summing the square root of the eigenvalues (i.e. a spectral function).

Denote $A = Z^{(1)\prime} Z^{(1)}$ and $B = Z^{(2)\prime} Z^{(2)}$. To show $h(\eta)$ is concave, we need to use the following trace convexity theorem (Petz, 1994, Proposition 2): Let $f : [0,\infty) \mapsto \mathbb{R}$ be a concave (convex) function, then $\mathrm{tr}(f(M))$ is concave (convex) on the positive semi-definite matrices M. Here $f(M)$ means applying l to the eigenvalues of M. In our case, take $f(x) = \sqrt{x}$ which is concave on $[0,\infty)$. Then for any $z_1, z_2 \in (0,1)$ and $\alpha_1, \alpha_2 > 0$ with $\alpha_1 + \alpha_2 = 1$, we have

$$\begin{aligned} h(\alpha_1\eta_1 + \alpha_2\eta_2) &= \mathrm{tr}\left[\left((\alpha_1\eta_1 + \alpha_2\eta_2)A + \frac{1 - (\alpha_1\eta_1 + \alpha_2\eta_2)}{\beta_2} B \right)^{1/2} \right] \\ &= \mathrm{tr}\left[\left(\alpha_1(\eta_1 A + \frac{1-\eta_1}{\beta_2} B) + \alpha_2(\eta_2 A + \frac{1-\eta_2}{\beta_2} B) \right)^{1/2} \right] \\ &\geq \alpha_1 \,\mathrm{tr}\left[(\eta_1 A + \frac{1-\eta_1}{\beta_2} B)^{1/2} \right] + \alpha_2 \,\mathrm{tr}\left[(\eta_2 A + \frac{1-\eta_2}{\beta_2} B)^{1/2} \right] \\ &= \alpha_1 h(\eta_1) + \alpha_2 g(\eta_2), \end{aligned}$$

where the $\geq$ used the trace convexity theorem. So g is concave.

Notice that $\|D_\rho^{-1}\Gamma\|_{tr}$ is only quasi-concave (which is why the change of variables is needed), because it is the composition of the concave function, $h(\eta)$ and the quasi-concave linear-fractional transformation, $\eta(\rho) = \frac{\beta_1^2}{\beta_2^2\rho + \beta_1^2}$. ■

D.4 Derivation of the boosting recovery algorithm

Once an optimal reconstruction Z is obtained, we need to recover the optimal factors C and Φ that satisfy

$$C\Phi = Z \quad \|\Phi\|_{2,1} = \|Z\|^*_{(\mathcal{C}_{2,2},2)} \quad \text{and } C_{:,i} \in \mathcal{C}_{2,2} \text{ for all } i. \tag{D.8}$$

The strategy will be to first recover the optimal dual solution Γ given Z, then use Γ to recover Φ and C. Given Γ and possible set of recovery dictionaries $\mathbf{C}(\Gamma)$, the recovery problem will be reduced to finding a vector $\boldsymbol{\mu}$ and matrix C such that $\boldsymbol{\mu} \geq 0$, $C_{:,i} \in \mathbf{C}(\Gamma)$ for all i, and $C\,\mathrm{diag}(\boldsymbol{\mu})C'\Gamma = Z$. Then, the derivation of the oracle for generating $\mathbf{c}$ and $\boldsymbol{\mu}$ in a boosting algorithm is described, as well as a simplification of $\mathbf{C}(\Gamma)$ that makes the oracle more efficient.

D.4.1 Characterizing the recovery set

From Lemma 6, we know that the regularizer on Φ can be expressed in terms of Γ:

$$\min_{\substack{\{C_{:,i}\in\mathcal{C}_{2,2}\\ \Phi:C\Phi=Z\}}} \|\Phi\|_{2,1} = \|Z'\|^*_{(\mathcal{C}_{2,2},2)} = \max_{\{\Gamma:\|D_\rho\Gamma\|_{sp}\le 1\}} \operatorname{tr}(\Gamma' Z) = \max_{\{\Gamma:\|\tilde{\Gamma}\|_{sp}\le 1\}} \operatorname{tr}(\tilde{\Gamma}' D_\rho^{-1} Z) = \max_{\rho\ge 0} \left\|D_\rho^{-1} Z\right\|_{tr}$$

For an optimal ρ and Z, Γ that satisfies $\|D_\rho\Gamma\|_{sp} = 1$ and $\operatorname{tr}(\Gamma' Z) = \left\|D_\rho^{-1} Z\right\|_{tr}$ is the corresponding optimal Γ. Let $U\Sigma V'$ be the SVD of $D_\rho^{-1} Z$ and set $\Gamma = D_\rho^{-1} UV'$. Then Γ satisfies

$$\begin{aligned}
&\|D_\rho\Gamma\|_{sp} = \left\|D_\rho D_\rho^{-1} UV'\right\|_{sp} = \left\|UV'\right\|_{sp} = 1 \quad \triangleright \text{because } U \text{ and } V \text{ are orthonormal}\\
&\operatorname{tr}(\Gamma' Z) = \operatorname{tr}(VU' D_\rho^{-1} Z) = \operatorname{tr}(U'(U\Sigma V')V) \quad \triangleright \text{because } \operatorname{tr}(ABC) = \operatorname{tr}(BCA) = \operatorname{tr}(CAB)\\
&\qquad = \operatorname{tr}(\Sigma) = \|D_\rho^{-1} Z\|_{\mathrm{tr}}.
\end{aligned}$$

Given such an optimal Γ, can characterize an optimal solution (C, Φ) using the set

$$\mathbf{C}(\Gamma) := \arg\max_{\mathbf{c}\in\mathcal{C}_{2,2}} \|\Gamma'\mathbf{c}\| = \left\{ \mathbf{c} = \begin{bmatrix} \mathbf{c}^{(1)} \\ \mathbf{c}^{(2)} \end{bmatrix} : \|\mathbf{c}^{(1)}\| = \beta_1, \|\mathbf{c}^{(2)}\| = \beta_2, \|\Gamma'\mathbf{c}\| = 1 \right\}. \tag{D.9}$$

Theorem 12. *For a dual optimal Γ, (C, Φ) solves recovery problem (D.8) if and only if $C_{:,i} \in \mathbf{C}(\Gamma)$ and $\Phi_{i,:} = \|\Phi_{i,:}\|_2 C'_{:,i}\Gamma$, such that $C\Phi = Z$.*

Proof: " $\Longrightarrow$ " If (C, Φ) solve the recovery problem (D.8) then $C\Phi = Z$ and $\|\Phi\|_{2,1} = \|Z\|^*_{(\mathcal{C}_{2,2},2)}$, giving

$$\|\Phi\|_{2,1} = \|Z\|^*_{(\mathcal{C}_{2,2},2)} = \operatorname{tr}(\Gamma' Z) = \operatorname{tr}(\Gamma' C\Phi) = \sum_i \Phi_{i,:}\Gamma' C_{:,i}. \tag{D.10}$$

Now, for any $C_{:,i} \in \mathcal{C}_{2,2}$, we have

$$\begin{aligned}
&\|\Gamma' C_{:,i}\|_2 \le 1 \qquad \triangleright \text{because } \|\Gamma' C_{:,i}\|_2 \le \max_{\mathbf{c}\in\mathcal{C}_{2,2}} \|\Gamma'\mathbf{c}\|_2 = \|\Gamma\|_{(\mathcal{C}_{2,2},2)} \le 1\\
&\Phi_{i,:}\Gamma' C_{:,i} = \|\Phi_{i,:}\Gamma' C_{:,i}\|_2 \le \|\Phi_{i,:}\|_2 \|\Gamma' C_{:,i}\|_2 \le \|\Phi_{i,:}\|_2.
\end{aligned}$$

Therefore, $\sum_i \|\Phi_{i,:}\Gamma' C_{:,i}\| \le \sum_i \|\Phi_{i,:}\|_2 = \|\Phi\|_2$; but we also know that for optimal (C, Φ), $\sum_i \|\Phi_{i,:}\|_2 = \sum_i \|\Phi_{i,:}\Gamma' C_{:,i}\| \le \|\Phi_{i,:}\|_2 \|\Gamma' C_{:,i}\|_2$. Therefore, $\|\Gamma' C_{:,i}\|_2 = 1$ and

$$\Phi_{i,:}\Gamma' C_{:,i} = \|\Phi_{i,:}\|_2 \implies \Phi_{i,:}\Gamma' C_{:,i} C'_{:,i}\Gamma = \|\Phi_{i,:}\|_2 C'_{:,i}\Gamma \implies \Phi_{i,:}\|\Gamma' C_{:,i}\|_2 = \|\Phi_{i,:}\|_2 C'_{:,i}\Gamma.$$

Therefore, $C_{:,i} \in \mathbf{C}(\Gamma)$ and $\Phi_{i,:} = \|\Phi_{i,:}\|_2 C'_{:,i}\Gamma$.

" $\Longleftarrow$ " On the other hand, if $\|\Gamma' C_{:,i}\|_2 = 1$ and $\Phi_{i,:} = \|\Phi_{i,:}\|_2 C'_{:,i}\Gamma$ such that $Z = C\Phi$, then we have $\|Z\|^*_{(\mathcal{C}_{2,2},2)} = \sum_i \|\Phi_{i,:}\|_2$, implying the optimality of (C, Φ). ■

We can further simplify this recovery set, which will be useful for generating $\mathbf{c} \in \mathbf{C}(\Gamma)$. In fact, its dual problem has been stated in (D.6). Once we obtain the optimal ρ, it is straightforward to backtrack and recover the optimal λ_1 and λ_2 in (D.6), because $\rho = \lambda_2/\lambda_1$ and $\beta_1^2\lambda_1 + \beta_2^2\lambda_2 = \|D_\rho\Gamma\|^2_{sp} = 1$, giving $\lambda_2 = \rho\lambda_1$ and $\lambda_1 = 1/(\beta_1^2 + \rho\beta_2^2)$.

Theorem 13. *Given* $R = \lambda_1 I_1 + \lambda_2 I_2 - \Gamma\Gamma'$ *for dual optimal* Γ *and optimal Lagrange multipliers* λ_1, λ_2, $r = \operatorname{rank}(R)$, $N = \begin{bmatrix} N_1 \\ N_2 \end{bmatrix} \in \mathbb{R}^{n\times(n-r)}$ *an orthonormal basis of the null-space of* R *and eigen-decomposition* $U\Sigma U' = \beta_2^2 N_1' N_1 - \beta_1^2 N_2' N_2$, *then*

$$\mathbf{C}(\Gamma) = \left\{ NU\mathbf{v} : \mathbf{v}'\Sigma\mathbf{v} = 0,\ \|\mathbf{v}\|^2 = \beta_1^2 + \beta_2^2 \right\}. \tag{D.11}$$

Proof: $\mathbf{C}(\Gamma)$ is the set of optimal solutions to

$$\max_{\mathbf{c}\in\mathcal{C}_{2,2}} \left\|\Gamma'\mathbf{c}\right\| \tag{D.12}$$

with an equivalent dual problem (D.6) described above:

$$\max_{\mathbf{c}\in\mathcal{C}_{2,2}} \left\|\Gamma'\mathbf{c}\right\| = \min_{\{\lambda_1\geq 0,\, \lambda_2\geq 0,\, \lambda_1 I_1 + \lambda_2 I_2 \succeq \Gamma\Gamma'\}} \beta_1^2\lambda_1 + \beta_2^2\lambda_2.$$

By the KKT conditions (Rockafellar, 1970, §28), $\mathbf{c}$ is an optimal solution to (D.12) if and only if

$$\left\|\mathbf{c}^{(1)}\right\| = \beta_1, \quad \left\|\mathbf{c}^{(2)}\right\| = \beta_2, \tag{D.13}$$

$$\left\langle R, \mathbf{c}\mathbf{c}'\right\rangle = \mathbf{0}, \ \text{ where } R = \lambda_1 I_1 + \lambda_2 I_2 - \Gamma\Gamma' \succeq 0. \tag{D.14}$$

First, (D.14) holds iff $\mathbf{c}$ is in the null space of R. To enforce this constraint, let $R = \tilde{U}\tilde{\Sigma}V'$ and use standard orthonormal basis for the null-space, $\{\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n\}$ (see end of Appendix D.1). Define

$$\mathbf{c} = N\boldsymbol{\alpha}, \quad \text{where} \quad N = [\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n] = \begin{bmatrix} N_1 \\ N_2 \end{bmatrix}, \ \boldsymbol{\alpha} \in \mathbb{R}^{n-r}. \tag{D.15}$$

Second, (D.13) is satisfied iff

$$0 = \beta_2^2 \left\|\mathbf{c}^{(1)}\right\|^2 - \beta_1^2 \left\|\mathbf{c}^{(2)}\right\|^2 = \boldsymbol{\alpha}' \left(\beta_2^2 N_1' N_1 - \beta_1^2 N_2' N_2\right) \boldsymbol{\alpha} \tag{D.16}$$

$$\beta_1^2 + \beta_2^2 = \|\mathbf{c}\|^2. \tag{D.17}$$

Now we simplify using several linear transformations. Perform eigen-decomposition $U\Sigma U' = \beta_2^2 N_1' N_1 - \beta_1^2 N_2' N_2$, where $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_{n-r})$, and $U \in \mathbb{R}^{(n-r)\times(n-r)}$ is orthonormal. Let $\mathbf{v} = U'\boldsymbol{\alpha}$. Therefore, we get new conditions on $\mathbf{v}$ to satisfy the conditions (D.13) and D.14 on $\mathbf{c}$,

$$\mathbf{c} = N\boldsymbol{\alpha} = NU\mathbf{v} \qquad \triangleright \text{ by (D.15), making } \mathbf{c} \text{ satisfy (D.14)}$$

$$\boldsymbol{\alpha}' U\Sigma U'\boldsymbol{\alpha} = \mathbf{v}'\Sigma\mathbf{v} = \sum_i \sigma_i v_i^2 = 0 \qquad \triangleright \text{ iff (D.16)} \tag{D.18}$$

$$\beta_1^2 + \beta_2^2 = \|\mathbf{c}\|^2 = \mathbf{v}'U'N'NU\mathbf{v} = \mathbf{v}'\mathbf{v} \qquad \triangleright \text{ iff (D.17), jointly satisfying (D.13).} \tag{D.19}$$

In summary, $\mathbf{c} = NU\mathbf{v}$ satisfies (D.12) for the following set

$$\mathbf{C}(\Gamma) = \{NU\mathbf{v} : \mathbf{v} \text{ satisfies (D.18) and (D.19)}\} = \left\{ NU\mathbf{v} : \mathbf{v}'\Sigma\mathbf{v} = 0,\ \|\mathbf{v}\|^2 = \beta_1^2 + \beta_2^2 \right\}. \ \blacksquare$$

Summary: given Γ, the recovery problem (D.8) has been reduced to finding a vector $\boldsymbol{\mu}$ and matrix C such that $\boldsymbol{\mu} \geq 0$, $C_{:,i} \in \mathbf{C}(\Gamma)$ for all i, and $C \operatorname{diag}(\boldsymbol{\mu}) C'\Gamma = Z$.

D.4.2 Boosting procedure

In this section, we demonstrate how to incrementally recover $\boldsymbol{\mu}$ and C. Denote the range of $C\,\mathrm{diag}(\boldsymbol{\mu})C'$ by the set

$$\mathbf{S} := \{\textstyle\sum_i \mu_i \mathbf{c}_i \mathbf{c}_i' : \mathbf{c}_i \in \mathbf{C}(\Gamma), \boldsymbol{\mu} \geq 0\}\,.$$

which is the conic hull of *possibly infinitely many* rank one matrices $\{\mathbf{c}\mathbf{c}' : \mathbf{c} \in \mathbf{C}(\Gamma)\}$. By Carathéodory's theorem (Rockafellar, 1970, §17), any matrix $M \in \mathbf{S}$ can be written as the conic combination of *finitely* many rank one matrices of the form $\{\mathbf{c}\mathbf{c}' : \mathbf{c} \in \mathbf{C}(\Gamma)\}$. Conceptually, therefore, the recovery problem has been reduced to finding a sparse set of non-negative weights, $\boldsymbol{\mu}$, over the set of feasible basis vectors, $\mathbf{c} \in \mathbf{C}(\Gamma)$.

To find these weights, we use a totally corrective "boosting" procedure that is guaranteed to converge to a feasible solution. Consider the following objective function for boosting

$$l(M) = \|M\Gamma - Z\|_F^2, \text{ where } M \in \mathbf{S}.$$

Note that l is clearly a convex function in M with a Lipschitz continuous gradient. Theorem 12 proves that an optimal recovery of corresponds precisely to those $M \in \mathbf{S}$ such that $l(M) = 0$. The idea behind totally corrective boosting is to find a minimizer of l (i.e., optimal recovery) incrementally. After initializing $M_0 = 0$, we iterate between two steps:

1. Weak learning step: find

$$\mathbf{c}_k \in \operatorname*{argmin}_{\mathbf{c}\in\mathbf{C}(\Gamma)} \left\langle \nabla l(M_{k-1}), \mathbf{c}\mathbf{c}' \right\rangle = \operatorname*{argmax}_{\mathbf{c}\in\mathbf{C}(\Gamma)} \mathbf{c}' Q \mathbf{c}, \tag{D.20}$$

where $Q = -\nabla l(M_{k-1}) = 2(Z - M_{k-1}\Gamma)\Gamma'$.

2. "Totally corrective" step:

$$\begin{aligned} \boldsymbol{\mu}^{(k)} &= \operatorname*{argmin}_{\boldsymbol{\mu}:\mu_i\geq 0} f\left(\textstyle\sum_{i=1}^k \mu_i \mathbf{c}_i \mathbf{c}_i'\right), \\ M_k &= \textstyle\sum_{i=1}^k \mu_i^{(k)} \mathbf{c}_i \mathbf{c}_i'. \end{aligned} \tag{D.21}$$

Three key facts can be established about this boosting procedure: (i) each weak learning step can be solved efficiently; (ii) each totally corrective weight update can be solved efficiently; and (iii) $l(M_k) \searrow 0$, hence a feasible solution can be arbitrarily well approximated. (ii) is immediate because (D.21) is a standard quadratic program and (iii) has been proved in (Zhang et al., 2012). Only (i) deserves some explanation. We derive an efficient algorithm for the oracle in the next subsection, using the simplified recovery set $\mathbf{C}(\Gamma)$.

D.4.3 Solving the weak oracle problem

The weak oracle needs to solve

$$\max_{\mathbf{c}\in\mathbf{C}(\Gamma)} \mathbf{c}'Q\mathbf{c},$$

where $Q = -\nabla l(M_{k-1}) = 2(Z - M_{k-1}\Gamma)\Gamma'$. By (D.11), this optimization is equivalent to

$$\begin{aligned}
\max_{\mathbf{c}\in\mathbf{C}(\Gamma)} \mathbf{c}'Q\mathbf{c} &= \max_{\mathbf{v}:\mathbf{v}'\Sigma\mathbf{v}=0,\ \|\mathbf{v}\|^2=\beta_1^2+\beta_2^2} \mathbf{v}'T\mathbf{v} && \triangleright \text{ where } T = U'N'QNU \\
&= \max_{\mathbf{v}:\mathbf{v}'\mathbf{v}=1,\mathbf{v}'\Sigma\mathbf{v}=0} \mathbf{v}'T\mathbf{v} \\
&= \max_{\Phi\succeq\mathbf{0},\mathrm{tr}(\Phi)=1,\mathrm{tr}(\Sigma\Phi)=0} \mathrm{tr}(T\Phi) && \triangleright \text{ if we let } \Phi = \mathbf{v}\mathbf{v}' \\
&= \min_{\tau,\omega:\tau\Sigma+\omega I-T\succeq\mathbf{0}} \omega && \triangleright \text{ by using the Lagrange dual} \\
&= \min_{\tau\in\mathbb{R}} \lambda_{\max}(T - \tau\Sigma) && \triangleright \text{ where } \lambda_{\max} \text{ is the maximum eigenvalue.}
\end{aligned}$$

Since $\lambda_{\max}$ is a convex function over real symmetric matrices, the last line search problem is convex in τ, hence can be solved globally and efficiently.

Given the optimal τ and the optimal objective value ω, the optimal $\mathbf{v}$ can be recovered using a similar approach used to simplify the set $\mathbf{C}(\Gamma)$. Let the null space of $\omega I + \tau\Sigma - T$ be spanned by $\hat{N} = \{\hat{\mathbf{n}}_1, \ldots, \hat{\mathbf{n}}_s\}$. Then find any $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^s$ such that $\mathbf{v} := \hat{N}\hat{\boldsymbol{\alpha}}$ satisfies $\|\mathbf{v}\|^2 = \beta_1^2 + \beta_2^2$ and $\mathbf{v}'\Sigma\mathbf{v} = 0$. The complete algorithm is given in Algorithm 3.

Appendix E

Efficient training for multi-view subspace learning

We show that the change of variables, $Q = E_\eta Z$, does not affect the concavity of the inner optimization in the maximin problem (5.3). Let $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$. To show that the change of variables does not affect the concavity of the function $g(\eta) = \min_Q L(E_\eta Q; X) + \alpha \|Q\|_{tr}$, let Q^*_η be optimal for the given η: $Q^*_\eta = \operatorname{argmin}_Q L(E_\eta Q; X) + \alpha \|Q\|_{tr}$. Correspondingly let $Z^*_\eta = E_\eta Q^*_\eta$. Then

$$
\begin{aligned}
g(t\eta_1 + (1-t)\eta_2) &= L(E_{t\eta_1+(1-t)\eta_2} Q^*_{t\eta_1+(1-t)\eta_2}; X) + \alpha \left\| Q^*_{t\eta_1+(1-t)\eta_2} \right\|_{tr} \\
&= L(Z^*_{t\eta_1+(1-t)\eta_2}; X) + \alpha \left\| E^{-1}_{t\eta_1+(1-t)\eta_2} Z^*_{t\eta_1+(1-t)\eta_2} \right\|_{tr} \\
&\geq L(Z^*_{t\eta_1+(1-t)\eta_2}; X) + \qquad \triangleright \text{ because } \left\| E_\eta^{-1} Z \right\|_{tr} \text{ is concave in } \eta \\
&\qquad t\alpha \left\| E^{-1}_{\eta_1} Z^*_{t\eta_1+(1-t)\eta_2} \right\|_{tr} + (1-t)\alpha \left\| E^{-1}_{\eta_2} Z^*_{t\eta_1+(1-t)\eta_2} \right\|_{tr} \\
&= t \left[L(Z^*_{t\eta_1+(1-t)\eta_2}; X) + \alpha \left\| E^{-1}_{\eta_1} Z^*_{t\eta_1+(1-t)\eta_2} \right\|_{tr} \right] + \\
&\qquad (1-t) \left[L(Z^*_{t\eta_1+(1-t)\eta_2}; X) + \alpha \left\| E^{-1}_{\eta_2} Z^*_{t\eta_1+(1-t)\eta_2} \right\|_{tr} \right] \\
&\geq t \operatorname*{argmin}_Z \left[L(Z; X) + \alpha \left\| E^{-1}_{\eta_1} Z \right\|_{tr} \right] + \\
&\qquad (1-t) \operatorname*{argmin}_Z \left[L(Z; X) + \alpha \left\| E^{-1}_{\eta_2} Z \right\|_{tr} \right] \\
&= t \left[L(Z^*_{\eta_1}; X) + \alpha \left\| E^{-1}_{\eta_1} Z^*_{\eta_1} \right\|_{tr} \right] + (1-t) \left[L(Z^*_{\eta_2}; X) + \alpha \left\| E^{-1}_{\eta_2} Z^*_{\eta_2} \right\|_{tr} \right] \\
&= t g(\eta_1) + (1-t) g(\eta_2).
\end{aligned}
$$

Therefore, $g(\eta)$ is concave in η.

Notice, however, that the new optimization no longer satisfies strong duality, and the max and min can no longer be swapped: for any fixed Q, the maximum η is 0 or 1 to send the loss to infinity. This means that $g(\eta)$ is concave in η, but $L(E_\eta Q; X) + \alpha \|Q\|_{tr}$ (without the min over Q) is not.

Appendix F

Semi-supervised learning appendix

This appendix contains the derivation of the hard and soft clustering algorithms for unsupervised and semi-supervised learning. The last section contains explicit derivations of the forward and reverse loss functions for the transfers used in the semi-supervised learning experiments.

F.1 Algorithms for clustering

Hard clustering: To obtain Algorithms 8 and 10, we prove the following lemmas.

Lemma 8. *For* $Y \in \{0,1\}^{k\times T}$, $\mathbf{1}Y = \mathbf{1}$,

$$D_{F^\star}(CY||\mathbf{f}(X)) = D_F(X||\mathbf{f}^\star(CY)) = D_F(X||\mathbf{f}^\star(C)Y)$$

Proof: From Lemma 2, we know that $D_F(X||\mathbf{f}^\star(CY)) = D_{F^\star}(CY||\mathbf{f}(X))$. Because $Y \in \{0,1\}^{k\times T}$ and $Y\mathbf{1} = \mathbf{1}$, we can see that CY simply selects columns of C, i.e. if there is a one at position $1 \leq i \leq k$, then column i in C is selected. Therefore, $\mathbf{f}^\star(CY) = \mathbf{f}^\star(C)Y$ and we conclude that $D_F(X||\mathbf{f}^\star(CY)) = D_F(X||\mathbf{f}^\star(C)Y)$. ■

We can now optimize over M for $D_F(X||MY)$. We can further simplify this objective, due to the fact that Y is discrete.

Lemma 9. *For a given* $Y \in \{0,1\}^{k\times T}$, $\mathbf{1}Y = \mathbf{1}$, $X \in$ *Dom*$(\mathbf{f})$ *and class* i,

$$\underset{M\in Dom(\mathbf{f})}{\operatorname{argmin}} \sum_{t:Y_{it}=1} D_F(X_{:t}||M_{:i}) = \frac{1}{Y_{i:}\mathbf{1}} \sum_{t:Y_{it}=1} X_{:t}$$

Proof: Let n_i be the number of instances with class i and $\mathbf{m} = M_{:i} \in \mathbb{R}^n$.

$$\frac{1}{n_i}\sum_{t:Y_{it}=1} D_F(X_{:t}||\mathbf{m}) = \frac{1}{n_i}\sum_{t:Y_{it}=1} F(X_{:t}) - F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top (X_{:t} - \mathbf{m})$$

$$= \bar{F} - \frac{1}{n_i}\sum_{t:Y_{it}=1} F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top \frac{1}{n_i}\sum_{t:Y_{it}=1} (X_{:t} - \mathbf{m}) \qquad \triangleright\ \bar{F} = \frac{1}{n_i}\sum_{t:Y_{it}=1} F(X_{:t})$$

$$= \bar{F} - F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top (\bar{\mathbf{x}} - \mathbf{m}) \qquad \triangleright\ \bar{\mathbf{x}} = \frac{1}{n_i}\sum_{t:Y_{it}=1} X_{:t}$$

$$= \bar{F} - F(\bar{\mathbf{x}}) + D_F(\bar{\mathbf{x}}||\mathbf{m}) \qquad \triangleright\ D_F(\bar{\mathbf{x}}||\mathbf{m}) = F(\bar{\mathbf{x}}) - F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top (\bar{\mathbf{x}} - \mathbf{m})$$

Therefore, we get

$$\min_{\mathbf{m}\in\mathbb{R}^{n\times k}} \sum_{t:Y_{it}=1} D_F(X_{:t}||\mathbf{m}) = \min_{\mathbf{m}\in\mathbb{R}^{n\times k}} n_i\bar{F} - n_i F(\bar{\mathbf{x}}) + n_i D_F(\bar{\mathbf{x}}||\mathbf{m}) = \min_{\mathbf{m}\in\mathbb{R}^{n\times k}} D_F(\bar{\mathbf{x}}||\mathbf{m}).$$

Since the Bregman divergence is guaranteed to be greater than or equal to zero, the minimum value for $D_F(\bar{\mathbf{x}}||\mathbf{m})$ is zero, obtained by setting $\mathbf{m} = \bar{\mathbf{x}}$. Therefore, for each instance t, the optimal setting for the inner minimization of $M_{:i} = \frac{1}{n_i}\sum_{t:Y_{it}=1} X_{:t}$. ■

From Lemmas 8 and 9, we obtain get the simplifications used in Equations (6.4) and (6.6) for Bregman hard clustering with non-linear transfers.

$$\begin{aligned}
\min_{\Phi\in\mathcal{F}} \min_{C\in Dom(F^\star)} D_{F^\star}(C\Phi||\mathbf{f}(X)) &= \min_{\Phi\in\mathcal{F}} \min_{M\in Dom(f)\subset\mathbb{R}^{n\times k}} D_F(X||M\Phi) \\
&= \min_{\Phi\in\mathcal{F}} \min_{M\in Dom(f)} \sum_{i=1}^{k} \sum_{t:\Phi_{it}=1} D_F(X_{:t}||M_{:i}) \\
&= \min_{\Phi\in\mathcal{F}} \sum_{i=1}^{k} \frac{1}{\Phi_{i:}\mathbf{1}} \sum_{t:\Phi_{it}} X_{:t}
\end{aligned}$$

Soft clustering: To obtain Algorithms 9 and 11, we will first prove the following lemma to characterize the optimization.

Lemma 10. *For* $Y \in [0,1]^{k\times T}$, $\mathbf{1}Y = \mathbf{1}$,

$$\min_{C\in\mathbb{R}^{n\times k}} \sum_{t=1}^{T} D_F(X_{:t}||\mathbf{f}^{-1}(CY_{:,t})) \leq \min_{M\in Dom(f)} \sum_{t=1}^{T}\sum_{i=1}^{k} Y_{it} D_F(X_{:t}||M_{:i})$$

Proof: First, by the convexity of the Bregman divergence in the first parameter,

$$\begin{aligned}
\sum_{t=1}^{T} D_F(X_{:t}||\mathbf{f}^{-1}(CY_{:t})) &= \sum_{t=1}^{T} D_{F^\star}(CY_{:t}||\mathbf{f}(X_{:t})) && \triangleright \text{ from Lemma 2} \\
&= \sum_{t=1}^{T} D_{F^\star}\left(\sum_{i=1}^{k} C_{:,i}Y_{it}\middle|\middle|\mathbf{f}(X_{:t})\right) \\
&\leq \sum_{t=1}^{T}\sum_{i=1}^{k} Y_{it} D_{F^\star}(C_{:i}||\mathbf{f}(X_{:t})) && \triangleright \text{ because convex in first parameter} \\
&= \sum_{t=1}^{T}\sum_{i=1}^{k} Y_{it} D_F(X_{:t}||\mathbf{f}^{-1}(C_{:i})) && \text{and } Y_{:t}\mathbf{1} = 1,\ Y_{it} \geq 0 \text{ for all } i,t
\end{aligned}$$

■

Now we can characterize the mixture model clustering optimization. We simplify the inner optimization first, assuming a fixed outer $\Phi \in \mathcal{F}$:

$$\begin{aligned}
&\min_{C\in\mathbb{R}^{n\times k}} \sum_{t=1}^{T}\sum_{i=1}^{k} D_F(X_{:t}||\mathbf{f}^{-1}(C\Phi_{:t})) \\
&\leq \min_{M\in\text{Dom}(f)} \sum_{t=1}^{T}\sum_{i=1}^{k} D_F(X_{:t}||M_{:i})\Phi_{it} && \triangleright \text{ by Lemma 10} && \text{(F.1)} \\
&\equiv \min_{M\in\text{Dom}(f)} -\sum_{t=1}^{T} \log\left(\sum_{i=1}^{k} p_i \exp(-D_F(X_{:t}||M_{:i}))\right) && \triangleright \text{ from Section 5 (Banerjee et al., 2005).}
\end{aligned}$$

We can also add a smoothness parameter ρ, because $\rho D_F(X_{:t}||M_{:i})$ is still a convex loss function. We obtain the final optimization

$$\min_{\mathbf{p}\geq 0,\mathbf{1}\mathbf{p}=1} \min_{M\in\text{Dom}(f)} \sum_{t=1}^{T} -\log\left(\sum_{i=1}^{k} p_i \exp(-\rho D_F(X_{:t}||M_{:i}))\right) \qquad \text{(F.2)}$$

where as $\rho \to \infty$, the objective approaches the hard clustering objective. Note that the ρ parameter only influences the update for Φ; the update for M is unaffected by ρ.

Finally, in Lemma 11, we prove that the inner minimization over M simplifies to an expectation and we get the updates shown in Algorithm 9.

Lemma 11. *For a given $Y \in [0,1]^{k\times T}$, $\mathbf{1}Y = \mathbf{1}$, $X \in Dom(\mathbf{f})$ and any $\rho > 0$,*

$$\operatorname*{argmin}_{M\in Dom(f)} -\sum_{t=1}^{T} \log\left(\sum_{i=1}^{k} p_i \exp(-\rho D_F(X_{:t}||M_{:i}))\right) = \left\{M_{:i} = \frac{1}{Y_{i:}\mathbf{1}}\sum_{t=1}^{T} Y_{it}X_{:t}\right\}$$

Proof: From (F.1), we know that

$$\operatorname*{argmin}_{M\in Dom(f)} -\sum_{t=1}^{T} \log\left(\sum_{i=1}^{k} p_i \exp(-\rho D_F(X_{:t}||M_{:i}))\right) = \operatorname*{argmin}_{M\in Dom(f)} \sum_{t=1}^{T}\sum_{i=1}^{k} Y_{it} D_F(X_{:t}||M_{:i}).$$

Fix class i. Then, using a similar argument as in Lemma 8, we will see that the optimal solution for $M_{:i}$ will be the mean vector, $\bar{\mathbf{x}} = (Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}X_{:t}$.

$$
\begin{aligned}
&(Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}D_F(X_{:t}||\mathbf{m}) \\
&= (Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}[F(X_{:t}) - F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top(X_{:t}-\mathbf{m})] \qquad \triangleright \text{ let } \bar{F} = (Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}F(X_{:t}) \\
&= \bar{F} - (Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top(Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}(X_{:t}-\mathbf{m}) \\
&= \bar{F} - F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top(\bar{\mathbf{x}}-\mathbf{m}) \qquad \triangleright \text{ as } (Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it} \text{ and } \bar{\mathbf{x}} = (Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}X_{:t} \\
&= \bar{F} - F(\bar{\mathbf{x}}) + D_F(\bar{\mathbf{x}}||\mathbf{m}) \qquad \triangleright D_F(\bar{\mathbf{x}}||\mathbf{m}) = F(\bar{\mathbf{x}}) - F(\mathbf{m}) - \mathbf{f}(\mathbf{m})^\top(\bar{\mathbf{x}}-\mathbf{m}).
\end{aligned}
$$

Therefore, we get

$$
\begin{aligned}
\min_{\mathbf{m}\in\mathbb{R}^{n\times k}} \sum_{t=1}^{T} Y_{it}D_F(X_{:t}||\mathbf{m}) &= \min_{\mathbf{m}\in\mathbb{R}^{n\times k}} (Y_{i:}\mathbf{1})^{-1}\bar{F} - (Y_{i:}\mathbf{1})^{-1}F(\bar{\mathbf{x}}) + (Y_{i:}\mathbf{1})^{-1}D_F(\bar{\mathbf{x}}||\mathbf{m}) \\
&= \min_{\mathbf{m}\in\mathbb{R}^{n\times k}} D_F(\bar{\mathbf{x}}||\mathbf{m})
\end{aligned}
$$

which has solution $\mathbf{m} = \bar{\mathbf{x}} = (Y_{i:}\mathbf{1})^{-1}\sum_{t=1}^{T} Y_{it}X_{:t}$, completing the proof. ■

Semi-supervised extensions: The semi-supervised forms of these algorithms arise simply from modifying the M-Step to use both the imputed and given labels. The E-step remains unchanged, but is implicitly affected by the fact that the choice of M is influenced by the labeled data.

F.2 Bregman divergences for standard semi-supervised learning

This section includes explicit derivations of the potential functions and transfers used for the standard semi-supervised learning experiments. Note that softmax is not invertible: for $\boldsymbol{\xi}(\mathbf{z}) = \mathbf{e}^{\mathbf{z}}/(\mathbf{1}^\top\mathbf{e}^{\mathbf{z}})$, $\boldsymbol{\xi}(\mathbf{z}) = \boldsymbol{\xi}(\mathbf{z} - z\mathbf{1})$ for any scalar z. If we add the restriction that $z_k = 0$, then we obtain invertibility (formally shown in the below proposition).

Proposition 14 (Softmax Invertibility)**.** *Let* $\boldsymbol{\xi}(\mathbf{z}) = \mathbf{e}^{\mathbf{z}}/(\mathbf{1}^\top\mathbf{e}^{\mathbf{z}})$. *The inverse map for* $\boldsymbol{\xi} : (\mathbb{R}^{k-1}, 0) \to S_k$ *with* $S_k = \{\mathbf{y} : \mathbf{y} \geq \mathbf{0},\ \mathbf{y}^\top\mathbf{1} = 1\}$ *exists and is* $\boldsymbol{\xi}^{-1}(\mathbf{y}) = \ln(\mathbf{y}) - \ln(y_k)\mathbf{1}$

Proof: Define a function $\mathbf{g} : S_k \to (\mathbb{R}^{k-1}, 0)$ as $\mathbf{g}(y) = \ln(\mathbf{y}) - \ln(y_k)\mathbf{1}$. Notice that this function is well-defined on S_k and the range is a subset of $(\mathbb{R}^{k-1}, 0)$ because the last element is always

zeroed by $\mathbf{g}$. Take any $\mathbf{z} \in (\mathbb{R}^{k-1}, 0)$. Then,

$$\begin{aligned}
\mathbf{g}(\boldsymbol{\xi}(\mathbf{z})) &= \ln(\boldsymbol{\xi}(\mathbf{z})) - \ln(\xi_k(\mathbf{z}))\mathbf{1} \\
&= \ln(\mathbf{e}^{\mathbf{z}}/(\mathbf{1}^\top \mathbf{e}^{\mathbf{z}})) - \ln(\mathbf{e}^0/(\mathbf{1}^\top \mathbf{e}^{\mathbf{z}}))\mathbf{1} \\
&= \ln\left(\frac{\mathbf{e}^{\mathbf{z}}}{\mathbf{1}^\top \mathbf{e}^{\mathbf{z}}}\right) + \ln\left(\left(\frac{1}{\mathbf{1}^\top \mathbf{e}^{\mathbf{z}}}\right)^{-1}\right)\mathbf{1} \\
&= \ln\left(\frac{\mathbf{e}^{\mathbf{z}}}{\mathbf{1}^\top \mathbf{e}^{\mathbf{z}}} \circ (\mathbf{1}^\top \mathbf{e}^{\mathbf{z}})\right) && \triangleright \text{ where } \circ \text{ is component-wise multiply} \\
&= \ln(\mathbf{e}^{\mathbf{z}}) = \mathbf{z}
\end{aligned}$$

Since $\mathbf{g} \cdot \boldsymbol{\xi}$ is the identity and the inverse function is unique, we know that $\boldsymbol{\xi}^{-1} = \mathbf{g}$. ■

Table F.1: Transfer functions with their inverses and potential functions.

	$f(x)$	$f^{-1}(y)$	$F(\mathbf{x})$	$F^\star(\mathbf{y})$
IDENTITY	$\mathbf{x}$	$\mathbf{x}$	$\mathbf{x}^2/2$	$\mathbf{y}^2/2$
SIGMOID	$\sigma(x) =(1+e^{-\mathbf{x}})^{-1}$	$\ln(\mathbf{y}/(\mathbf{1}-\mathbf{y}))$	$\mathbf{1}'\ln(\mathbf{1}+e^{\mathbf{x}})$	$\mathbf{y}\ln(\mathbf{y}/(\mathbf{1}-\mathbf{y})) + \mathbf{1}\ln(\mathbf{1}-\mathbf{y})$
SOFTMAX	$\xi(x) = \ e^{\mathbf{x}}/\mathbf{1}'e^{\mathbf{x}}$	$\ln(\mathbf{y}) - \ln(\mathbf{y}_k)\mathbf{1}$	$\ln(\mathbf{1}'e^{\mathbf{x}})$	$[\ln(\mathbf{y}) - \ln(\mathbf{y}_k)\mathbf{1}]\mathbf{y} - \ln(\mathbf{1}'(\mathbf{y} - \mathbf{y}_k\mathbf{1}))$
EXP	$e^{\mathbf{x}}$	$\ln(\mathbf{y})$	$\mathbf{1}'e^{\mathbf{x}}$	$[\ln(\mathbf{y}) - \mathbf{1}]\mathbf{y}'$
CUBE	$\mathbf{x}^3$	$\mathbf{x}^{1/3}$	$\mathbf{1}'\mathbf{x}^4/4$	$\mathbf{y}^{1/3}\mathbf{y}' - 0.25\mathbf{y}^{4/3}\mathbf{1}$

Appendix G

Autoregressive Moving Average Models

G.1 Proof of Lemma 1 and 4

Proof: A standard argument follows (Neal and Hinton, 1998). First note that a lower bound on $\log p(\mathbf{x}_{1:T}|\Theta)$ can be easily obtained:

$$\begin{aligned}
\log p(\mathbf{x}_{1:T}|\Theta) &= \log \int p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)\, d\boldsymbol{\varepsilon}_{1:T} \\
&= \log \int q(\boldsymbol{\varepsilon}_{1:T}) \frac{p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)}{q(\boldsymbol{\varepsilon}_{1:T})}\, d\boldsymbol{\varepsilon}_{1:T} \quad \text{for any density } q(\cdot),\ q(\boldsymbol{\varepsilon}_{1:T}) > 0 \text{ everywhere} \\
&\geq \int q(\boldsymbol{\varepsilon}_{1:T}) \left(\log \frac{p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)}{q(\boldsymbol{\varepsilon}_{1:T})} \right) d\boldsymbol{\varepsilon}_{1:T} \quad \text{by Jensen's inequality (since } \log \text{ is concave)} \\
&= \int q(\boldsymbol{\varepsilon}_{1:T}) \log p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)\, d\boldsymbol{\varepsilon}_{1:T} - \int q(\boldsymbol{\varepsilon}_{1:T}) \log q(\boldsymbol{\varepsilon}_{1:T})\, d\boldsymbol{\varepsilon}_{1:T} \\
&= \int q(\boldsymbol{\varepsilon}_{1:T}) \log p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)\, d\boldsymbol{\varepsilon}_{1:T} + H(q(\cdot)). \qquad \text{(G.1)}
\end{aligned}$$

It remains to show that the maximization of the lower bound attains the original value; that is:

$$\log p(\mathbf{x}_{1:T}|\Theta) = \max_{q(\cdot)} \int q(\boldsymbol{\varepsilon}_{1:T}) \log p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)\, d\boldsymbol{\varepsilon}_{1:T} + H(q(\cdot))$$

over densities $q(\cdot)$. This can be verified merely by choosing the particular density $q(\boldsymbol{\varepsilon}_{1:T}) = p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta)$ and verifying that

$$\begin{aligned}
\text{(G.1)} &= \int p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta) \log p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)\, d\boldsymbol{\varepsilon}_{1:T} - \int p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta) \log p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta)\, d\boldsymbol{\varepsilon}_{1:T} \qquad \text{(G.2)} \\
&= \int p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta) \log \frac{p(\mathbf{x}_{1:T}, \boldsymbol{\varepsilon}_{1:T}|\Theta)}{p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta)}\, d\boldsymbol{\varepsilon}_{1:T} \\
&= \int p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta) \log p(\mathbf{x}_{1:T}|\Theta)\, d\boldsymbol{\varepsilon}_{1:T} \\
&= \log p(\mathbf{x}_{1:T}|\Theta) \int p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta)\, d\boldsymbol{\varepsilon}_{1:T} \\
&= \log p(\mathbf{x}_{1:T}|\Theta), \qquad \text{(G.3)}
\end{aligned}$$

implying that the upper bound can always be attained by $q(\boldsymbol{\varepsilon}_{1:T}) = p(\boldsymbol{\varepsilon}_{1:T}|\mathbf{x}_{1:T}, \Theta)$. ■

G.2 Proof of Theorems 9 and 10

Lemma 12. *Given a convex loss function $L(\cdot, \mathbf{x})$ and convex regularizer $R(\cdot)$ with $\gamma \geq 0$, the following loss is convex in Z for all $\mathbf{x}_t \in \mathbb{R}^n$*

$$L_A\left(\sum_{j=0}^{q} Z^{(j)}{}_{:,t-j}, \mathbf{x}_t\right) = L\left(\sum_{j=0}^{q} Z^{(j)}{}_{:,t-j}, + \sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i}, \mathbf{x}_t\right) + \gamma R(A)$$

Proof: Let $g_t((Z, A)) = \sum_{j=0}^{q} Z^{(j)}_{:,t-j} + \sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i}$. We need to show that $L(g_t((Z, A)), \mathbf{x}_t)$ is convex for any $\mathbf{x}_t$.

1. Clearly $g_t((Z, A))$ is convex in (Z, A), because it is a linear function of the two variables.
2. Since $L(\cdot, \mathbf{x}_t)$ is convex, and the composition of two convex functions is convex, then $L(g_t(\cdot), \mathbf{x}_t)$ is convex. ■

Corollary 6. *The parameter estimation problem with a convex regularizer $R(\cdot)$ on the autoregressive parameters, A,*

$$\min_A \min_Z \sum_{t=\max(p,q)}^{T} L\left(Z^{(j)}_{:,t-j} + \sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i}, \mathbf{x}_t\right) + \alpha\|Z\| + \gamma R(A) \tag{G.4}$$

is jointly convex in A and Z for $\alpha, \gamma \geq 0$.

Now, since $L_{A,t}(\cdot)$ is convex, we know that we can apply Corollary 3 from Chapter 4.

G.3 Generalizations for regularized ARMA modeling

There are several clear generalizations to ARMA models that are important for practical applications. These include the addition of exogenous input variables (ARMAX), the generalization to non-stationary series (ARIMA) and generalizing the set of possible regularizers chosen on $\boldsymbol{\varepsilon}_t$. This section describes these three extensions.

Regularized ARMAX models We can trivially add exogenous variables because, like the autoregressive component, they are included in the loss additively:

$$L\left(\sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i} + \sum_{j=0}^{q} Z^{(j)}_{:,t-j} + \sum_{i=1}^{s} C^{(i)}\mathbf{u}_{t-i}\ ;\ \mathbf{x}_t\right)$$

where $\mathbf{u}_t \in \mathbb{R}^d$ is an input control vector or exogenous vector. As with the autoregressive component, we can add a convex regularizer on $C \in \mathbb{R}^{n \times ds}$ to avoid overfitting. The resulting optimization is an alternation over the three parameters, A, Z and C.

Regularized ARIMA models This generalization is similarly simple, because an autoregressive integrated moving average, ARIMA(p,d,q), model is simply an ARMA(p,q) model of the time series differenced d times. Differencing is a form of taking the derivative, with the assumption that the time lag is appropriately small. As a result, the more times the differencing is applied, the more likely we are to reach a stationary distribution.

Regularized ARMA models with different regularizers The convex formulation for RARMA was completed under a Frobenius norm regularizer on $\boldsymbol{\varepsilon}_t$, to encode the typical Gaussian prior on the innovations. This distribution, however, can be changed to $\|\mathcal{E}\|_{2,1}$ based on the discussion before Corollary 3. Below, we show that this block $2,1$-norm corresponds to assuming a prior on the innovations that is a Laplacian distribution across time.

There are several multivariate extensions of Laplace distributions; we choose a multivariate Laplace, parametrized by a mean, $\boldsymbol{\mu}_i$, and scatter matrix Σ_i, with the convenient pdf (Arslan, 2010):

$$p_L(\mathcal{E}_{i,:}|\boldsymbol{\mu}_i, \Sigma_i) = \frac{|\Sigma_i|^{-1/2}}{2^T \pi^{\frac{T-1}{2}} \Gamma(\frac{T+1}{2})} e^{-\sqrt{(\mathcal{E}_{i,:}-\boldsymbol{\mu}_i)\Sigma_i^{-1}(\mathcal{E}_{i,:}-\boldsymbol{\mu}_i)'}}$$

As before, where the covariance was pushed into the B parameters, we assume $\boldsymbol{\mu} = \mathbf{0}$ and $\Sigma = I$, giving

$$\begin{aligned}
&-\log p_L(\mathcal{E}_{i,:}|\boldsymbol{\mu}_i, \Sigma_i) = \\
&\qquad \frac{1}{2}\log\left(|\Sigma_i|\right) + T\log(2) + \frac{T-1}{2}\log(\pi) + \log\Gamma\left(\frac{T+1}{2}\right) + \sqrt{(\mathcal{E}_{i,:}-\boldsymbol{\mu}_i)\Sigma_i^{-1}(\mathcal{E}_{i,:}-\boldsymbol{\mu}_i)'} \\
&\implies \min_{\mathcal{E}} \sum_{i=1}^{k} -\log p_L(\mathcal{E}_{i,:}|\boldsymbol{\mu}_i = \mathbf{0}, \Sigma_i = I) = \min_{\mathcal{E}} \sum_{i=1}^{k} \sqrt{\mathcal{E}_{i,:}\mathcal{E}_{i,:}'} = \min_{\mathcal{E}} \|\mathcal{E}\|_{2,1}
\end{aligned}$$

We can now simplify the relationship between the hidden variables because this multivariate Laplace distribution decomposes nicely into the multiplication of a scalar gamma-distributed variable, $S_i \sim G(\frac{t+1}{2}, \beta = \frac{1}{2})$ the covariance matrix $\Sigma \in \mathbb{R}^{t\times t}$ and independent, standard normal variables, $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, I)$ (Arslan, 2010):

$$\mathcal{E}_{i,:} = \sqrt{S_i}\Sigma\boldsymbol{\epsilon}_i \qquad \text{where} \qquad p_{S_i}(s) = \frac{1}{\Gamma(\frac{t+1}{2})2^{\frac{t+1}{2}}} s^{\frac{t-1}{2}} \exp\left(-\frac{s}{2}\right).$$

Interestingly, this makes the connection with the Frobenius norm formulation more clear, since once the scale is fixed, we have independent innovations. The scalar across time acts like a shared scale on the covariance of the innovations.

In general, there are potentially many other distributional assumptions we can make on the innovation variables that could be efficiently solvable, depending on advances in convex reformulations of matrix factorization.

Algorithm 14 ARMA synthetic data generation

Input: p, q, dimension of series n, number of samples T
Output: A, B, $\mathbf{x}_1, \ldots, \mathbf{x}_T$

$m \leftarrow n/p$ // Size of partition in $\mathbf{x}$
$s \leftarrow 0.999$ // Scale $s < 1$
$\tau \leftarrow 3$ // Permutation period
for $i = 1 : p$ **do**
  $\mathbf{d} \leftarrow []$ // Eigenvalues of permutation matrix $\tilde{A}^{(i)}$
  if m is odd **then**
    $\mathbf{d} \leftarrow s$
    count $\leftarrow$ count +1
  end if
  while count $< m$ **do**
    dnew $\leftarrow \exp(2\pi\sqrt{-1}/\tau)$
    $\mathbf{d} \leftarrow [\mathbf{d}$ dnew conj(dnew)$]$
    $\mathbf{v}_1 \leftarrow$ randn$(p, 1), \mathbf{v}_1 \leftarrow$ randn$(p, 1)$
    $V = [V,\ \mathbf{v}_1 + \sqrt{-1}\mathbf{v}_2,\ \mathbf{v}_1 - \sqrt{-1}\mathbf{v}_2]$
    count $\leftarrow$ count +2
  end while
  $A^{(i)} = \mathbf{0}$
  $A^{(i)}_{(ip+1:ip+m),(ip+1:ip+m)} = \text{real}(VDV^{-1})$
end for
$B \leftarrow$ randn(qn, n)
$B_{:j} \leftarrow B_{:j} / \sum_{i=1}^{qn} B_{ij}^2$
// Simulate data from generated A, B and $\mathcal{E}$
$(\mathbf{x}_1, \ldots, \mathbf{x}_T) \leftarrow$ simulate$(A, B, \mathcal{E})$
return $A, B, (\mathbf{x}_1, \ldots, \mathbf{x}_T)$

G.4 Details for the algorithms and experiments

G.4.1 Generating stable synthetic ARMA models

For the autoregressive part, we need to choose the parameters $A^{(i)}$ carefully, otherwise the system will be unstable and the generated sequences will diverge (Lütkepohl, 2007). For vector ARMA in particular, there are not many approaches to generating stable ARMA models and most experiments involve small, known systems. We use an approach where each $A^{(i)}$ acts as a permutation matrix on a sub-part of $\mathbf{x}$. The observation $\mathbf{x}$ is partitioned into p sub-vectors of size $m = n/p$. Each $A^{(i)}$ permutes the ith block in $\mathbf{x}$, with a slow decay for numerical stability. Therefore, $A^{(i)}$ has zeros in entries corresponding to blocks $1, \ldots, i-1$ and $i+1, \ldots, n$ and a slowly decaying permutation matrix at row and columns entries $ip+1$ to $ip+m$. This permutation matrix is generated using randomly generated orthonormal vectors $\mathbf{v}$ and diagonal D, and the conjugates of each eigenvalue and eigenvector, to give $\tilde{A}^{(i)} = VDV^{-1}$. The decaying rate s_0 is set to 0.999 for numerical sta-

bility and the period of the permutation matrix determine by τ. See Algorithm 14 for the complete generation procedure. The initial $\mathbf{x}_{-p}, \ldots, \mathbf{x}_0$ are sampled from the unit ball in $\mathbb{R}^n$.

For the moving average part, the entries of $B^{(j)}$ and $\boldsymbol{\varepsilon}_t$ are drawn from standard normal, $\mathcal{N}(\mathbf{0}, I)$. The matrix B is normalized to have unit variance.

G.4.2 Imputing future innovations efficiently

In practice, parameters A and B are typically learned on training data $\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}$ and then fixed. Given these fixed parameters and the imputed $\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_{t-1}$, we may wish to impute innovation variables for future data, $\mathbf{x}_t$:

$$\begin{aligned}
&\min_{\boldsymbol{\varepsilon}_{:,t}} L\left(\sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i} + \sum_{j=1}^{q} B^{(j)}\boldsymbol{\varepsilon}_{t-j} + B^{(0)}\boldsymbol{\varepsilon}_t, \mathbf{x}_t\right) + \frac{\alpha}{2}\left\|\boldsymbol{\varepsilon}_{1:t}\right\|_F^2 \\
&\quad = \min_{\boldsymbol{\varepsilon}_t} \; L\left(\sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i} + \sum_{j=1}^{q} B^{(j)}\boldsymbol{\varepsilon}_{t-j} + B^{(0)}\boldsymbol{\varepsilon}_t, \mathbf{x}_t\right) + \frac{\alpha}{2}\left(\sum_{i=1}^{k}\sum_{j=1}^{t-1} \boldsymbol{\varepsilon}_{i,j}^2 + \boldsymbol{\varepsilon}_{i,t}^2\right) \\
&\quad = \min_{\boldsymbol{\varepsilon}_t} L\left(\sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t-i} + \sum_{j=1}^{q} B^{(j)}\boldsymbol{\varepsilon}_{t-j} + B^{(0)}\boldsymbol{\varepsilon}_t, \mathbf{x}_t\right) + \frac{\alpha}{2}\|\boldsymbol{\varepsilon}_t\|_2^2
\end{aligned}$$

This procedure was not necessary for these prediction experiments; however, it could be useful for predictions that are updated over time.

G.4.3 Forecasting in ARMA models

A main use of ARMA models is for forecasting, once the model parameters have been estimated. In particular, given the parameters, Θ, we would like to predict the value of a future observation, $\mathbf{x}_{t+h}$, given observations of a history $\mathbf{x}_1, \ldots, \mathbf{x}_t$. Under the Gaussian assumption (and exponential family distributions more generally, see Appendix G.3) the optimal point predictor, $\hat{\mathbf{x}}_{t+h}$, is given by the conditional expectation

$$\tilde{\mathbf{x}}_{t+h} = E[\mathbf{x}_{t+h}|\mathbf{x}_1, \ldots, \mathbf{x}_t, \Theta], \tag{G.5}$$

which can be easily computed from the observed history and the parameters. To understand ARMA forecasting in a little more detail, first consider the one step prediction case. If the innovation variables are included in the observed history, then the one step conditional expectation is easily determined to be

$$\hat{\mathbf{x}}_{t+1} = E[\mathbf{x}_{t+1}|\mathbf{x}_1, \ldots, \mathbf{x}_t, \boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_t, \Theta] = \sum_{i=1}^{p} A^{(i)}\mathbf{x}_{t+1-i} + \sum_{j=1}^{q} B^{(j)}\boldsymbol{\varepsilon}_{t+1-j}. \tag{G.6}$$

For the h step forecast, since the expected innovations for $\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+h-1}$ are zero given previous innovations, we obtain

$$\hat{\mathbf{x}}_{t+h} = E[\mathbf{x}_{t+h}|\mathbf{x}_1, \ldots, \mathbf{x}_t, \boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_t, \Theta] = \sum_{i=1}^{p} A^{(i)} \hat{\mathbf{x}}_{t+h-i} + \sum_{j=h}^{q} B^{(j)} \boldsymbol{\varepsilon}_{t+h-j} \tag{G.7}$$

where for $h > q$, the moving average term is no longer used, and $\hat{\mathbf{x}}_{t+h-j} = \mathbf{x}_{t+h-j}$ for $t+h-j \leq t$.

If, however, the innovation variables $\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_t$ are not observed, then $\mathbf{x}_{t+1}$ becomes dependent on the entire history $\mathbf{x}_1, \ldots, \mathbf{x}_t$. The one step conditional expectation then becomes

$$\tilde{\mathbf{x}}_{t+1} = E[\mathbf{x}_{t+1}|\mathbf{x}_1, \ldots, \mathbf{x}_t, \Theta] = \sum_{i=1}^{p} A^{(i)} \mathbf{x}_{t+1-i} + \sum_{j=1}^{q} \tilde{B}^{(j)}_{(t)} \tilde{\boldsymbol{\varepsilon}}_{t+1-j} \tag{G.8}$$

where $\tilde{\boldsymbol{\varepsilon}}_t = \mathbf{x}_t - \tilde{\mathbf{x}}_t$, and the $\tilde{B}^{(j)}_{(t)}$ can be efficiently computed, recursively, from the previous $\tilde{B}^{(k)}_{(n)}$ and the original parameters; see for example (Brockwell and Davis, 2002, Sec. 3.3) for details. This leads to the optimal h step predictor that can be efficiently computed using the same recursively updated quantities

$$\tilde{\mathbf{x}}_{t+h} = E[\mathbf{x}_{t+h}|\mathbf{x}_1, \ldots, \mathbf{x}_t, \Theta] = \sum_{i=1}^{p} A^{(i)} \tilde{\mathbf{x}}_{t+h-i} + \sum_{j=h}^{t+h-1} \tilde{B}^{(j)}_{(t+h-1)} \tilde{\boldsymbol{\varepsilon}}_{t+h-j}. \tag{G.9}$$