THE UNIVERSITY OF CHICAGO

APPLICATIONS OF ADAPTIVE SHRINKAGE IN MULTIPLE STATISTICAL PROBLEMS

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF STATISTICS

BY

WEI WANG

CHICAGO, ILLINOIS
DECEMBER 2017

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my adviser, Professor Matthew Stephens, for his superb guidance, endless support and limitless encouragement over the years. His support, guidance, insight and erudition have been invaluable and have provided a phenomenal research environment in our research laboratory. It has been truly my great honor to work with such an extraordinary statistician and geneticist.

I am grateful to my committee members, John Lafferty and Tracy Ke, for their inspiring suggestion, insightful discussion and constructive feedback on the topics in my research. Their input has been greatly helpful to improve this dissertation.

I would like to thank other faculty members in the Departments of Statistics and Human Genetics: Michael Stein, Steven Lalley, Mary Sara McPeek, Dan Nicolae, Rina Foygel Barber, John Novembre, Xin He and others, for their generous help in my research and study.

I would also like to thank my friends and academic peers in s in the Department of Statistics and the Stephens Lab: Yuancheng Zhu, Liwen Zhang, Marc Goessling, Jacob Degner, Zhe Liu, Zhengrong Xin, Jiajun Shen, Heejung Shim, Gao Wang, David Gerard, Mengyin Lu, John Zekos, Hussein Al-Asadi, Kushal Dey, Siming Zhao, Xiang Zhou, John Blischak, Peter Carbonetto, Joe Marcus, Nicholas Knoblauch and many many more, for their encouragement, support and and intellectual conversations. It has been a great pleasure to work with so many friendly, creative and talented people in such a collaborative environment.

Finally, I wish to express the profound gratitude to my beloved parents, Ailian

and Zengrong, for their unconditional encouragement, love and support. I also wish to thank my girlfriend, Dong Yan, for her love, support and sacrifice.

ABSTRACT

Bayesian shrinkage method provides attractive approaches in large-scale data analysis with advantages in prediction and interpretation. Stephens (2016) presented a flexible Empirical Bayes method, "Adaptive Shrinkage" or *ash*, to the normal means problem. We explore the idea of adaptive shrinkage in multiple statistics problems, such as sparse factor analysis, sparse logistic factor analysis, estimation of precision matrix, multivariate regression and multiple testing with correlated observation.

In this dissertation, we propose a novel Empirical Bayes Factor Analysis (EBFA) and provide a framework of variational inference algorithm. We can apply any Empirical Bayes method, which provides first and second order moments, for normal means problem, and we focus on "Adaptive Shrinkage" method. Both factors and loadings have flexible shrinkage priors in this framework. We extend this EBFA method to logistic factor analysis on binary and binomial distributed data. We show that we can apply the same variational framework in this model by taking a tight lower bound of the likelihood function or augment the model with a Polya-Gamma random variable. We also apply the EBFA method to estimate the precision matrix and conduct a comparison of the methods based on sparse assumption and methods based on low rank assumption. We also introduce a hybrid procedure to combine these two types of methods. We provide a Variational Bayes (VB) algorithm for Bayesian sparse regression with a flexible prior. We also show that this VB approach can solve the normal means problem with correlated observation. Numerical studies on both real data and simulated data illustrate the efficacy of our methods.

CHAPTER 1

INTRODUCTION

Shrinkage is a powerful tool in statistical inference (Stein et al., 1956; James and Stein, 1961; Efron and Morris, 1973). There are several ways to implementing shrinkage in practice. One particularly attractive way is via Empirical Bayes (EB) methods. One key feature of EB methods is that they directly address a key issue in shrinkage analysis, which is *how much to shrink*. Specifically they do this by estimating a prior distribution for the quantities that are to be shrunk (see below for some simple examples). Different EB approaches differ from one another in their choice of (usually parametric) family for this prior distribution. Common choices include, for example, i) a normal distribution; ii) a mixture of a point mass at 0 and a double exponential distribution Johnstone et al. (2004); or iii) a mixture of a point mass at 0 and a normal distribution Clyde and George (2000); George and Foster (2000). Recently, Stephens (2016) introduced an EB approach to shrinkage that is based on a more flexible semi-parametric families of distributions – for example, the family of distributions that are unimodal at 0.

Stephens (2016) presented methods to fit these flexible EB methods, which is called "Adaptive Shrinkage" or *ash*, to the simplest shinkage problem: the normal means problem (below). In this dissertation, we explore applications of these ideas to several other problems where shrinkage is desirable: sparse factor analysis, "generalized" sparse factor analysis, multiple regression, a generalized normal means problem with correlated errors, and precision matrix estimation. We illustrate in many examples how the more flexible semi-parametric priors in *ash* lead to improved inference

compared with other approaches.

1.0.1 The EB normal means (EBNM) model

The following simple problem, which we call the "Empirical Bayes normal means" (EBNM) problem, plays a key role in our work.

Suppose we have observations $\boldsymbol{x} = (x_1, \ldots, x_n)$ of underlying quantities $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n)$, with Gaussian errors with known standard deviations $\boldsymbol{s} = (s_1, \ldots, s_n)$. That is,

$$\boldsymbol{x} = \boldsymbol{\theta} + \boldsymbol{e} \tag{1.0.1}$$

where $e_i \sim N(0, s_i^2)$. Suppose further that the elements of $\boldsymbol{\theta}$ are assumed independent and identically distributed (i.i.d.) from some distribution, $g \in \mathcal{G}$, where $\mathcal{G}$ is some specified family of distributions:

$$\theta_1, \ldots, \theta_n \sim^{iid} g, \quad g \in \mathcal{G}. \tag{1.0.2}$$

One motivation for this last assumption is that, by choosing $\mathcal{G}$ appropriately, it can capture the idea that $\boldsymbol{\theta}$ may be "sparse", with many elements at or near 0. For example, Johnstone et al. (2004) consider g to be a mixture of a point mass at 0 and a double exponential (Laplace) distribution; Clyde and George (2000) use a mixture of a point mass at 0 and a zero-mean Gaussian distribution (sometimes known as "spike-and-slab" distributions). Here we use the more flexible families of unimodal distributions from Stephens (2016); see "Adaptive shrinkage" below.

The Empirical Bayes approach to fitting the model (1.0.1)-(1.0.2) proceeds in two

steps:

1. Estimate g by maximum likelihood:

$$\hat{g} = \arg\max_{g \in \mathcal{G}} L(g), \tag{1.0.3}$$

   where

$$L(g) := p(\boldsymbol{x}|g) = \prod_j \int p(x_j|\theta_j, s_j) g(d\theta_j). \tag{1.0.4}$$

2. Estimate θ_j using its posterior distribution given $\hat{g}$,

$$p(\theta_j|\boldsymbol{x}, \boldsymbol{s}, \hat{g}) \propto \hat{g}(\theta_j) p(x_j|\theta_j, s_j). \tag{1.0.5}$$

   For example, we could estimate θ_j using the mean of this posterior distribution.

We call this two-step procedure "solving the EB normal means (EBNM) problem". Formally, the procedure defines a mapping from the known quantities $(\boldsymbol{x}, \boldsymbol{s})$ (and the family $\mathcal{G}$) to $(\hat{g}, \boldsymbol{p})$ where $\hat{g}$ is given in (1.0.3) and $\boldsymbol{p}$ denotes the posterior distribution on $\boldsymbol{\theta}$ given $\hat{g}$. Using *EBNM* to denote this mapping we can write:

$$EBNM(\boldsymbol{x}, \boldsymbol{s}) = (\hat{g}, \boldsymbol{p}). \tag{1.0.6}$$

Note that the posterior $\boldsymbol{p}$ is given by the product of the marginals in (1.0.5):

$$\boldsymbol{p}(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{s}, \hat{g}) = \prod_j p(\theta_j|\boldsymbol{x}, \boldsymbol{s}, \hat{g}). \tag{1.0.7}$$

1.0.2 Adaptive Shrinkage

Stephens (2016) introduces methods that he refers to as "adaptive shrinkage" (*ash*), which essentially solve the EBNM problem for several different choices of $\mathcal{G}$, including:

- $\mathcal{G} = \mathcal{U}$, the set of all unimodal distributions, with mode at 0;
- $\mathcal{G} = \mathcal{S}U$, the set of all symmetric unimodal distributions, with mode at 0;
- $\mathcal{G} = \mathcal{S}N$, the set of all scale mixtures of zero-centered normals.

In practice the methods work by approximating these families by finite mixture distributions with many components. The approximation can be made arbitrarily accurate by using increasingly many components. These methods are computationally stable and efficient, being based on convex optimization methods Koenker and Mizera (2010) and analytic Bayesian posterior computations.

Solving the EBNM problem is generally an attractive way to perform shrinkage estimation for $\boldsymbol{\theta}$ and/or induce sparsity in the estimates (Johnstone et al., 2004). In particular, by estimating g from the data, the EB approach learns *how much* shrinkage or sparsity is appropriate in a given context. Although all EB methods have this property to some extent, the unimodal families $\mathcal{G}$ underlying *ash* are particularly attractive, because their flexibility allows them to adapt to a wider range of sparsity patterns than more restrictive parametric families like spike-and-slab distributions.

1.1 Dissertation outline and contributions

In chapter 2, we extend the ideas underlying *ash* from the normal means problem to factor analysis (with normally-distributed errors). Factor analysis is a popular multivariate analysis method which describes the observed data as a linear superposition of latent factors. It is widely used in many areas including computational biology, social sciences, marketing, signal analysis, and finance . We develop methods that use a *ash* prior to flexibly induce sparsity on both latent factors and their corresponding loadings. We develop a variational EM algorithm to infer the parameters and hyperparameters of this model. We demonstrate the efficacy of our method by numerical studies of both real and simulated data. We also illustrate how this approach can help the interpretation of the factor model, recovering factors with different sparsity levels.

In chapter 3, we extend the sparse factor analysis method for normal data in chapter 2 to discrete (bernoulli and binomial) data. We do this by incorporating a logistic link function. The logistic factor analysis is to factor analysis as logistic regression is to linear regression. We again develop a variational EM algorithm to estimate the approximate posterior distribution for factors and loadings. We use numerical simulation studies to show the advantages of this approach for modeling binary and binomial data. And we show an application to genetic data that illustrates how the method can uncover bi-clustering structure in discrete data.

In chapter 4 we perform an extensive simulation study to compare several different approaches to estimating a large precision matrix from limited data. Estimating a precision matrix (and covariance matrix) is a fundamental statistical problem with

applications in many areas, such as computational biology, finance and social science. We compare two types of methods: those that assume the precision matrix is sparse, and those that are based on a factor model for the data matrix (equivalent to assuming that the precision is the sum of a low rank matrix and a diagonal matrix). We also introduce a hybrid procedure that combine these two types of method. We perform these simulations using a reproducible and extensible framework currently being developed in the Stephens lab, which is easy to add new method and new data sets without repeating efforts on existing results. It provides a reproducible and efficient benchmark system to compare methods for precision matrix estimation.

In chapter 5, we introduce a Bayesian approach to sparse regression with adaptive shrinkage prior. We introduce a Variational Bayes algorithm to estimate the hyper parameters of the prior distribution and approximate the posterior distribution. We add our method into an existing empirical benchmark for sparse regression problem to compare performance across different methods. We also show how this method can be used to solve a version of the normal means problem in which observations are correlated (with known correlation structure). This approach can be applied in multiple testing problem with known covariance structure. Numerical studies show the benefits from including covariance information.

In summary, this dissertation provides methods that extend the *ash* EB approach in Stephens (2016) to a range of important problems, and illustrates the benefits of this approach using both simulated and real data examples.

CHAPTER 2

FLASH: FACTOR AND LOADING ADAPTIVE SHRINKAGE

2.1 Introduction

Dimension reduction is an important technique in multivariate statistical analysis with a wide variety of applications in high dimensional data analysis. With simple geometric interpretations and computational complexity, dimension reduction is widely used in data compression, clustering, classification, pattern recognition and data visualization. Cunningham and Ghahramani (2015) provide a review of linear dimension reduction methods developed based on different motivations, such as principal component analysis (PCA, Pearson (1901)), factor analysis (Spearman, 1904), probabilistic PCA (Tipping and Bishop, 1999), sufficient dimensionality reduction (SDR, Adragni and Cook (2009)), independent component analysis (ICA, Hyvarinen et al. (2001)), canonical correlations analysis (CCA, Hotelling (1936)) and many other approaches.

Factor analysis is a statistical tool for dimension reduction which linearly maps high dimensional observed data points with p features across n samples into a lower dimensional subspace. It can be considered as a way to represent the data by a low-rank matrix and an independent noise, where the low-rank matrix is constructed by a linear combination of latent factors with different weights (loadings). Factor analysis has been widely used since first introduced by Spearman (1904) in handling large-scale data sets in in many areas (Engelhardt and Stephens, 2010; Price et al.,

2006; Leek and Storey, 2008; Bai and Ng, 2008; Lam et al., 2011). There are many large-scale data sets in which the relations among a large number of observed variables can be explained by a small number of latent factors, such as gene expression data, recommendation system, financial data, video data and speech data. Taking gene expression data as an example, the expression of thousands of genes in different tissues are co-regulated by a small number of underlying factors, such as transcription factor. In molecular biology and genetics, transcription of gene is affected by transcription factors binding to enhancer or promoter regions, which leads to different gene expression (up- or down-regulated). One transcription factor, as an activator or a repressor, regulates more than one genes. The gene regulatory network, which reflects the regulatory relationship between transcription factors and genes, is known to be sparse (Hartemink, 2005).

One challenge of factor analysis is the unidentifiability of latent structure. In factor analysis, the factors and loadings are invariant up to a rotation matrix. This identifiablity issue can be alleviated by introducing sparsity on factor matrix or loading matrix. Another challenge is the interpretation of the factors and loadings. Sparsity plays an important role in latent factor models, because it is easier to interpret with only a small number of variables and samples associated with the latent factors, and has a better performance in prediction due to only keeping representative elements in factors and loadings. In this article we mainly focus on sparse factor analysis. In practice, not all observed variables should be correlated with latent factors, and not all samples should be loaded on the latent factors either, so introducing sparsity on factors and loadings makes practical sense as well. It provides a regu-

larization mechanism to deal with identifiability issues in over-parameterized model, and reflects our prior belief on the true nature of the latent structure.

Sparsity can be introduced in many ways. Some penalized likelihood approaches have been developed through regularization via different types of penalty (Zou et al., 2006a; Witten et al., 2009; Johnstone and Lu, 2004). In a Bayesian context, we can put a sparsity inducing prior distribution to each element of factors. A general sparsity-inducing prior should have a point mass around zero to put strong shrinkage towards zero when signal is small, and also a heavy tail to keep the signal away from strong shrinkage when signal is large. Bayesian sparse factor analysis is first introduced for microarray gene expression data (West, 2003) and also applied to transcriptional regulatory networks (Sabatti and James, 2006), clusters of co-regulated genes (Pournara and Wernisch, 2007), biological pathway analysis (Carvalho et al., 2008), and population structure analysis (Engelhardt and Stephens, 2010).

Penalized likelihood method is one of the popular solutions. Recently, penalization likelihood methods have been deeply explored and widely used in many models and problems. l_1 penalty (LASSO regularization) has been applied in related works, such as sparse principal component analysis (Zou et al., 2006a) and penalized matrix decomposition (Witten et al., 2009). With the tuning parameter is chosen properly, the penalized methods get sparse estimations in factors or loadings. This approach is closely related to Bayesian approach with shrinkage prior which also leads to sparse solution. LASSO type regularization estimation corresponds to Maximum a Posterior (MAP) parameter estimation with Laplace distribution as prior. This type of methods lead to sparse parameters estimation but not sparse posterior distribution.

This property might be problematic if we want to incorporate prior information given that the true nature of the underlying structure is sparse. However, this approach has computational advantage by addressing the estimation problem over continuous parameter space.

Bayesian approach, as an alternative solution, introduces sparsity with different sparse inducing priors. Zero-norm prior, which puts certain probability to point mass at zero to achieve sparse solution for posterior distribution, has been used in factor model with Markov Chain Monte Carlo (MCMC) (West, 2003; Carvalho et al., 2008), but this approach is computationally expensive, since it needs to make inference over discrete parameters. Spike and slab prior, a mixture of normal distribution and point mass at zero, is widely used in different approaches (Sabatti and James, 2005; Knowles and Ghahramani, 2011; Hore et al., 2016). This prior leads to certain sparsity in posterior distribution by putting certain probability on the point mass at zero. Tail behavior is also important for prediction to fit the nonzero values. The variance of normal distribution from the "slab" part needs to be estimated properly, which is another challenge in the inference. We introduce a more flexible prior, adaptive shrinkage prior, to induce sparsity and shrinkage into factors and loadings.

In this chapter, we introduce a Bayesian approach to sparse factor analysis, based on using the adaptive shrinkage prior on both the factors and the loadings. The adaptive shrinkage prior we put here is a data-driven prior to determine the amount to shrink, which is more flexible. In this way, we can recover both sparse or dense factors and factors with a wide range of sparse level. Choosing proper rank of low dimensional structure is a challenge in this type of model. Model selection can be

a way to choose the value of rank, but a significant artificial threshold or searching range needs to be predetermined. In our work, we can determine the rank of the low-rank structure automatically from the inferred posterior distribution of factors and loadings. There is no explicit form for the parameters estimation, so MCMC could be one potential choice to estimate the parameters, which is however computationally intensive for large-scale data sets. We introduce a computationally convenient way using a variational EM algorithm to estimate the parameters and hyper parameters.

2.2 Sparse Factor Model

2.2.1 Regularization method

Factor analysis is a widely used dimension reduction method to represent relationship among a large number of variables by a small number of latent factors (features). Principal Component Analysis (PCA) is close related with factor analysis, which explain the data into a linear combination of latent factors. In PCA, with respect of the matrix decomposition, the data can be decomposed into:

$$Y = UDV^T \tag{2.2.1}$$

$$\hat{Y} = \sum_K d_k u_k v_k^T \tag{2.2.2}$$

where $U_{N\times P} = \{u_1, \cdots, u_P\}$, $V_{P\times P} = \{v_1, \cdots, v_P\}$ and $D = diag(d_1, \cdots, d_P)$. $\hat{Y}$ is an approximation of Y.

To make this decomposition identifiable, we need constraints on U and V, which

assume that they are orthonormal matrix. PCA, as a dimension reduction method, can be written as the solution of Maximum Likelihood Estimation (MLE) of a factor model which tries to explain the data by latent factors with certain assumptions (Tipping and Bishop, 1999). Here we call U as factors and V as loadings. To introduce sparsity to factors and loadings, we can put penalties on u_k and v_k. There are various types of penalties on u_k and v_k we could choose, such as Lasso type (l_1) penalty. We can also put different types of penalties on factors and loadings and even restrict factors and loadings to be non-negative for better interpretability. Penalized Matrix Decomposition (Witten et al., 2009) uses this regularized likelihood with the objective function:

$$\begin{aligned} & \min_{d,U,V} ||Y - U_K D_K V_K^T||_F^2 \\ \textit{subject to} \quad & ||U_k||_2^2 = ||V_k||_2^2 = 1, p_k^u(U_k) \leq c_k^u, p_k^v(V_k) \leq c_k^v, d_k \geq 0 \end{aligned} \tag{2.2.3}$$

where $k = 1, \cdots, K$, p_k^u and p_k^v are penalties we would like to put on the k^{th} column of U, U_k, or k^{th} column of V, V_k, and c_k^u and c_k^v are tuning parameters to control the sparsity level. There is no analytical solution for these tuning parameters, so we need to apply cross validation to get the optimal values of the parameters. We can see there are many parameters ($2 \times K$) to tune by cross validation, which is computational intensive. In large scale data, the intensive computation of cross validation for those tuning parameters might take too long time. It is because we need to fit the model every time for each given possible combination of candidate

grids of tuning parameters.

2.2.2 Bayesian method

In this chapter, we focus on Bayesian method. Starting with a generative model for factor analysis:

$$Y = LF^T + E \tag{2.2.4}$$

where Y is $N \times P$ matrix, L is $N \times K$ matrix, F is $P \times K$ matrix and E is $N \times P$ matrix and $E_{ij} \sim N(0, \sigma_{ij}^2)$, $i = 1, \cdots, N$ and $j = 1, \cdots, P$.

The sparse extension of the factor model is our main interest. Sparsity can be considered as a regularization mechanism to prevent overfitting model, and also the reflection of the prior belief for the true underlying structure of the data. However, in general it is unclear how sparse the factors and loadings should be. Thus, we need a model with flexible priors on both factors and loadings to answer three questions: 1, Shall we believe that there are sparsity structures on factors, loadings or both? 2, How sparse should we consider our prior information to be? 3, What kind of prior information shall we use for non-zero values?

2.2.3 More flexible prior for sparsity: Adaptive Shrinkage

We apply Adaptive Shrinkage (ASH) prior introduced by Stephens (2016), which naturally results in shrinkage estimation using a unimodal prior. We take one special

case as example:

$$g(\cdot;\pi) \quad = \quad \pi_0\delta_0(\cdot) + \sum_k \pi_k N(\cdot; 0, \sigma_k^2) \tag{2.2.5}$$

This prior is more general than spike and slab prior by using more flexible distribution to characterize the tail behavior. The grids of the variance for each component in the mixture of normal distributions are dense enough to characterize the tail behavior.

By estimating the π values from the data, the sparsity of the prior is estimated from (and so adaptive to) the data. As a data driven prior, it automatically determines the sparsity level and capture the tail behavior for the nonzero elements. In this way, we could provide a sparse posterior distribution if the data leans towards sparse factors and loadings, and provide a more reasonable and flexible shrinkage if the data leans to non-sparse factors. However the inference is still a challenge. In Bayesian inference, especially for the high dimensional data sets, MCMC methods tend to mix slowly even if we apply this unimodal prior. It is because of the multimodal property of the posterior distribution. Given the appealing properties to capture more general and flexible prior information from the data, it worths to explore a way to do Bayesian inference on this problem.

2.3 Empirical Bayes Factor Analysis (EBFA)

Here we extend the ideas behind the EBNM problem to factor analysis, yielding "Empirical Bayes Factor Analysis" (EBFA). While special cases of EBFA have appeared previously (e.g) we are not aware of a previous general formulation. Just

as the EB approach provides an attractive way to induce shrinkage in the normal means model, it also provides an attractive way to induce shrinkage on the factors and loadings in a Factor Analysis.

The computations for EBFA are considerably harder than those for EBNM. However, by exploiting variational approximations (Neal and Hinton, 1998; Blei et al., 2016), we derive an iterative algorithm to approximately solution to the EBFA problem where each iteration involves the same computations required to solve the EBNM problems. Variational approximations have been used in the past for fitting FA models (Stegle et al., 2012; Hore et al., 2016), and our methods here can be thought of as generalizing those approaches.

2.3.1 The single factor EBFA model

We start by defining a single factor ("rank 1") EBFA model; see Section 2.3.3 for the extension to multiple factors. The single factor EBFA model is:

$$Y = \boldsymbol{l}\boldsymbol{f}^T + E \tag{2.3.1}$$

$$l_1, \ldots, l_n \sim^{iid} g_{\boldsymbol{l}}, \quad g_{\boldsymbol{l}} \in \mathcal{G} \tag{2.3.2}$$

$$f_1, \ldots, f_p \sim^{iid} g_{\boldsymbol{f}}, \quad g_{\boldsymbol{f}} \in \mathcal{G} \tag{2.3.3}$$

$$E_{ij} \sim N(0, 1/\tau_{ij}) \text{ with } \boldsymbol{\tau} := (\tau_{ij}) \in \mathcal{T}. \tag{2.3.4}$$

Here Y is the $n \times p$ data matrix, $\boldsymbol{l}$ is an n-vector (the "loadings"), $\boldsymbol{f}$ is a p-vector (the "factor"), E is an $n \times p$ matrix of independent error terms, and $\boldsymbol{\tau}$ denotes the $n \times p$ matrix of precisions (τ_{ij}) which is assumed to lie in some space $\mathcal{T}$. (This can

be useful to impose some assumed structure on $\boldsymbol{\tau}$, such as column-specific precisions $\tau_{ij} = \tau_j$ for example.) As we discuss later, our methods allow that some elements of Y_{ij} could be "missing" (making the assumption that they are "missing at random") and indeed can estimate ("impute") missing values.

As with the EBNM model, there are many possible choices of distributional family $\mathcal{G}$, and this choice will affect the shrinkage on $\boldsymbol{l}$ and $\boldsymbol{f}$. (It would be easy to extend (2.3.1) to allow $\mathcal{G}$ to differ for factors and loadings, but for simplicity we use a single $\mathcal{G}$.) We can also extend the the model to deal with non-negative matrix factorization by setting $\mathcal{G}$ to be distribution family which only allows $\boldsymbol{l}$ and $\boldsymbol{f}$ to be non-negative.

Fitting the EBFA model involves estimating both the distributions $g_{\boldsymbol{l}}, g_{\boldsymbol{f}}$ and the values of $\boldsymbol{l}, \boldsymbol{f}$ (as well as $\boldsymbol{\tau}$). It would be natural, in principle, to do this as follows:

- Estimate $g_{\boldsymbol{l}}, g_{\boldsymbol{f}}$ and $\boldsymbol{\tau}$, by maximizing the likelihood:

$$L(g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) = \int \int p(Y|\boldsymbol{l}, \boldsymbol{f}, \boldsymbol{\tau})\, g_{\boldsymbol{l}}(d\boldsymbol{l})\, g_{\boldsymbol{f}}(d\boldsymbol{f}) \tag{2.3.5}$$

  over $g_{\boldsymbol{l}}, g_{\boldsymbol{f}} \in \mathcal{G}$ and $\boldsymbol{\tau} \in \Sigma$.

- Estimate $\boldsymbol{l}$ and $\boldsymbol{f}$ using their posterior distribution: $p(\boldsymbol{l}, \boldsymbol{f}|Y, \hat{g}_{\boldsymbol{l}}, \hat{g}_{\boldsymbol{f}}, \hat{\boldsymbol{\tau}})$.

However, both these two steps seem difficult, so we resort to variational approximations (Blei et al., 2016) which can be thought of as approximating this approach.

2.3.2 A Variational Approximation

To explain the variational approximation approach, begin by writing the log of the likelihood (2.3.11) as:

$$l(g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) := \log[p(Y|g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})] \tag{2.3.6}$$

$$= F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) + D_{KL}(q||p) \tag{2.3.7}$$

where

$$F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) = \int q(\boldsymbol{l}, \boldsymbol{f}) \log \frac{p(Y, \boldsymbol{l}, \boldsymbol{f}|g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})}{q(\boldsymbol{l}, \boldsymbol{f})} \, d\boldsymbol{l} \, d\boldsymbol{f} \tag{2.3.8}$$

and

$$D_{KL}(q||p) = -\int q(\boldsymbol{l}, \boldsymbol{f}) \log \frac{p(\boldsymbol{l}, \boldsymbol{f}|Y, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})}{q(\boldsymbol{l}, \boldsymbol{f})} \, d\boldsymbol{l} \, d\boldsymbol{f}. \tag{2.3.9}$$

This identity holds for any distribution $q(\boldsymbol{l}, \boldsymbol{f})$. Because the term D is non-negative, it follows that $F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})$ is a lower bound for the log likelihood:

$$l(g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) \geq F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) \tag{2.3.10}$$

with equality when $q(\boldsymbol{l}, \boldsymbol{f}) = p(\boldsymbol{l}, \boldsymbol{f}|Y, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})$.

In other words,

$$l(g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) = \max_{q} F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}), \tag{2.3.11}$$

where the maximization is over all possible distributions $q(\boldsymbol{l}, \boldsymbol{f})$. Maximizing $l(g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})$ can thus be viewed as maximizing F over $q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}$. However, as noted above, this maximization appears difficult. The variational approach simplifies the problem by

maximizing F but restricting the family of distributions for q. Specifically, let $\mathcal{Q}$ denote the family of distributions that "fully-factorize":

$$\mathcal{Q} = \left\{ q : q(\boldsymbol{l}, \boldsymbol{f}) = \prod_{i=1}^{n} q_{l,i}(l_i) \prod_{j=1}^{p} q_{f,j}(f_j) \right\}. \tag{2.3.12}$$

The variational approach seeks to optimize F over $q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}$ with the constraint $q \in Q$. For $q \in Q$ we can write $q(\boldsymbol{l}, \boldsymbol{f}) = q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f})$ where $q_{\boldsymbol{l}}(\boldsymbol{l}) = \prod_{i=1}^{n} q_{l,i}(l_i)$ and $q_{\boldsymbol{f}}(\boldsymbol{f}) = \prod_{j=1}^{p} q_{f,j}(f_j)$, and we can consider the problem as maximizing $F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})$.

Optimization Algorithm

We optimize $F(q_l, q_f, g_l, g_f, \boldsymbol{\tau})$ by iterating between optimizing over variables related to $\boldsymbol{l}$ (q_l, g_l), over variables related to $\boldsymbol{f}$ (q_f, g_f), and over $\boldsymbol{\tau}$. Each step is guaranteed to increase F, but F may be multi-modal, and there is no guarantee that it will reach a global optimum.

Algorithm 1 Variational Optimization for EBFA (rank 1)

1: $t = 0$

2: initialize $q_{\boldsymbol{l}}^0$, $q_{\boldsymbol{f}}^0$, $g_{\boldsymbol{l}}^{(0)}$, $g_{\boldsymbol{f}}^{(0)}$ and $\boldsymbol{\tau}^{(0)}$

3: $\epsilon = 1$

4: **while** $\epsilon > 10^{-5}$ **do**

5: $\quad t = t + 1$

6: $\quad q_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{l}}^{(t)} \leftarrow \arg\max_{q_{\boldsymbol{l}}, g_{\boldsymbol{l}}} F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}^{(t-1)}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}^{(t-1)}, \boldsymbol{\tau}^{(t-1)})$

7: $\quad q_{\boldsymbol{f}}^{(t)}, g_{\boldsymbol{f}}^{(t)} \leftarrow \arg\max_{q_{\boldsymbol{f}}, g_{\boldsymbol{f}}} F(q_{\boldsymbol{l}}^{(t)}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{f}}, \boldsymbol{\tau}^{(t-1)})$

8: $\quad \boldsymbol{\tau}^{(t)} \leftarrow \arg\max_{\boldsymbol{\tau}} F(q_{\boldsymbol{l}}^{(t)}, q_{\boldsymbol{f}}^{(t)}, g_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{f}}^{(t)}, \boldsymbol{\tau})$

9: $\quad \epsilon = F(q_{\boldsymbol{l}}^{(t)}, q_{\boldsymbol{f}}^{(t)}, g_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{f}}^{(t)}, \boldsymbol{\tau}^{(t)}) - F(q_{\boldsymbol{l}}^{(t-1)}, q_{\boldsymbol{f}}^{(t-1)}, g_{\boldsymbol{l}}^{(t-1)}, g_{\boldsymbol{f}}^{(t-1)}, \boldsymbol{\tau}^{(t-1)})$

return $list(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})$

The key steps in Algorithm 1 are the maximizations in Steps 6 and 7 (which are essentially the same, but switching the role of $\boldsymbol{l}$ and $\boldsymbol{f}$). A key result is that each of these steps essentially involves solving an EBNM problem. The following Proposition formalizes this for Step 6; a similar proposition holds for Step 7.

Proposition 1. *Step 6 in Algorithm 1 is solved by solving the EBNM problem. Specifically (dropping superscripts and subscripts involving iteration number t to simplify notation):*

$$(g_{\boldsymbol{l}}, q_{\boldsymbol{l}}) \leftarrow EBNM(\boldsymbol{x}, \boldsymbol{s}) \tag{2.3.13}$$

where

$$x_i = \frac{\sum_j \tau_{ij} Y_{ij} E_{q_{\boldsymbol{f}}}(f_j)}{\sum_{j \in \Gamma_i} \tau_{ij} E_{q_{\boldsymbol{f}}}(f_j^2)} \tag{2.3.14}$$

$$s_i^2 = \frac{1}{\sum_j \tau_{ij} E_{q_{\boldsymbol{f}}}(f_j^2)}. \tag{2.3.15}$$

Proof. See Appendix A.1. □

Missing data

If some elements of $Y_{i\cdot}$ are missing, then this is easily dealt with: the sums in (2.3.14) and (2.3.15) are simply computed using only the non-missing elements. (One way to implement this in practice is to set $\tau_{ij} = 0$ for (i, j) such that Y_{ij} is missing.)

Intuition

For intuition into where the EBNM in Proposition 1 comes from, consider estimating $\boldsymbol{l}, g_l$ in (2.3.1) *with* f *and* $\boldsymbol{\tau}$ *known*. The factor model then essentially becomes n independent regressions of the rows of Y on f, and for the ith regression a sufficient statistic for l_i is the maximum likelihood estimate:

$$\hat{l}_i = \frac{\sum_j \tau_{ij} Y_{ij} f_j}{\sum_j \tau_{ij} f_j^2}. \tag{2.3.16}$$

Further, it is easy to show that

$$\hat{l}_i \sim N(l_i, s_i^2) \tag{2.3.17}$$

where

$$s_i^2 = \frac{1}{\sum_j \tau_{ij} f_j^2}. \tag{2.3.18}$$

Combining (2.3.17) with the prior

$$l_1, \ldots, l_n \sim^{iid} g_l, \quad g_l \in \mathcal{G} \tag{2.3.19}$$

yields an EBNM problem.

The EBNM in Proposition 1 is essentially the same as the EBNM (2.3.17)-(2.3.19) , but with the terms f_j and f_j^2 replaced with their expectations under q_f. That is, the update for (q_l, g_l) in Algorithm 1, with $(q_f, g_f, \boldsymbol{\tau})$ fixed, is closely connected to solving the EBFA problem for "known $f, \boldsymbol{\tau}$".

2.3.3 The K-factor EBFA model

The single factor EBFA model naturally generalizes to allow for K factors:

$$Y = \sum_{k=1}^{K} \boldsymbol{l}_k \boldsymbol{f}_k^T + E \tag{2.3.20}$$

$$l_{ki} \sim^{iid} g_{l_k} \in \mathcal{G} \tag{2.3.21}$$

$$f_{kj} \sim^{iid} g_{f_k} \in \mathcal{G} \tag{2.3.22}$$

$$E_{ij} \sim N(0, 1/\tau_{ij}) \text{ with } \boldsymbol{\tau} := (\tau_{ij}) \in \mathcal{T}. \tag{2.3.23}$$

Extending the variational approximation to this model involves extending the

variational distributions q_1, q_f to be distributions on all K loadings/factors:

$$q_l(\boldsymbol{l}_1, \cdots, \boldsymbol{l}_K) = \prod_k q_{l_k}(\boldsymbol{l}_k) \tag{2.3.24}$$

$$q_f(\boldsymbol{f}_1, \ldots, \boldsymbol{f}_K) = \prod_k q_{f_k}(\boldsymbol{f}_k). \tag{2.3.25}$$

The objective function $F(q_l, q_f, g_l, g_f, \boldsymbol{\tau})$ now becomes a function of $q_l = (q_{\boldsymbol{l}_1}, \ldots, q_{\boldsymbol{l}_K})$, $q_f = (q_{\boldsymbol{f}_1}, \ldots, q_{\boldsymbol{f}_K})$, $g_l = (g_{l_1}, \ldots, g_{l_K})$ and $g_f = (g_{f_1}, \ldots, g_{f_K})$.

$$F(q_l, q_f, g_l, g_f, \boldsymbol{\tau}) = \int \prod_k q_{l_k}(\boldsymbol{l}_k) q_{f_k}(\boldsymbol{f}_k) \log \frac{p(Y, \boldsymbol{l}, \boldsymbol{f}; g_{l_1}, g_{f_1} \cdots g_{l_K} g_{f_K}, \boldsymbol{\tau})}{\prod_k q_{l_k}(\boldsymbol{l}_k) q_{f_k}(\boldsymbol{f}_k)} \tag{2.3.26}$$

For each k, we optimize F over the parts relating to factor k, first with respect to $q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}$ and then with respect to $q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}$ keeping other parts fixed. The optimizations are essentially identical to those for the rank 1 model, but with Y_{ij} replaced with the residuals obtained by removing the estimated effects of the other $k-1$ factors:

$$R^k_{ij} := Y_{ij} - \sum_{k' \neq k} \bar{l}_{ik'} \bar{f}_{k'j}. \tag{2.3.27}$$

Details are in Appendix A.3.

2.3.4 Estimating the precision

Consider now optimizing F over the precision parameters $\boldsymbol{\tau}$. Focusing on the parts of F that depend on $\boldsymbol{\tau}$ gives:

$$F(\boldsymbol{\tau}) = E_{q_{\boldsymbol{l}}} E_{q_{\boldsymbol{f}}} \sum_{ij} (1/2)\log(\tau_{ij}) - \tau_{ij}(Y_{ij} - l_i f_j)^2/2 + \text{const} \tag{2.3.28}$$

$$= (1/2)\sum_{ij} \log(\tau_{ij}) + \tau_{ij}\bar{R}_{ij}^2 + \text{const} \tag{2.3.29}$$

where

$$\bar{R}_{ij}^2 := \left\{ Y_{ij}^2 - 2Y_{ij} E_{q_{\boldsymbol{l}}} l_i E_{q_{\boldsymbol{f}}} f_j + E_{q_{\boldsymbol{l}}} l_i^2 E_{q_{\boldsymbol{f}}} f_j^2 \right\}. \tag{2.3.30}$$

These equations are for the single factor model, but are easy to extend to the K factor model.

If we constrain $\boldsymbol{\tau} \in \mathcal{T}$ then we have

$$\hat{\boldsymbol{\tau}} = \arg\max_{\boldsymbol{\tau} \in \mathcal{T}} \sum_{ij} [\log(\tau_{ij}) - \tau_{ij}\bar{R}_{ij}^2]. \tag{2.3.31}$$

Details are in Appendix A.5.

2.3.5 Factor and Loading Adaptive Shrinkage

In EBFA framework, apply any Empirical Bayes method, which provides first and second order moments, for normal means problem, and we focus on "Adaptive Shrinkage" method from Stephens (2016). We introduce this adaptive shrinkage prior to both factor and loading, so we call our method as Factor and Loading Adaptive

SHrinkage (FLASH). We apply R-package 'ashr' to solve the EBNM problem in FLASH method and we can easily get the objective function from the result of 'ashr'. Details of calculating the objective function are in Appendix A.2.

2.4 Simple Numerical Illustrations

We use simulated data to illustrate the flexibility of FLASH compared with several other existing approaches. Specifically we compare with KSVD (Aharon et al., 2006), SFAmix (Gao et al., 2013), SFA (Engelhardt and Stephens, 2010), NSFA (Knowles and Ghahramani, 2011), "Penalized Matrix Decomposition" (Witten et al., 2009) (implemented in the R package 'PMA') and "Sparse Singular Value Decomposition" (Yang et al., 2014) (SSVD, implemented in R package 'ssvd').

2.4.1 Single factor

We simulated data with $n = 200, p = 300$ according to the following single-factor model with sparse loadings, and a non-sparse factor:

$$Y = \boldsymbol{l}\boldsymbol{f}^T + E \tag{2.4.1}$$

where

$$f_j \sim N(0,1) \tag{2.4.2}$$

$$l_i \sim \pi_0\delta_0 + (1-\pi_0)\sum_{m=1}^{5}\frac{1}{5}N(0,\sigma_m^2) \tag{2.4.3}$$

$$E_{ij} \sim N(0,1/\tau) \tag{2.4.4}$$

where $(\sigma_1,\ldots,\sigma_5) = (0.25, 0.5, 1, 2, 4)$. We simulated two different scenarios on the loadings: strong sparsity ($\pi_0 = 0.9$) and intermediate sparsity ($\pi_0 = 0.3$). (We set the noise precision $\tau = 1, 1/16$ respectively in these two settings.) The true value $K = 1$ is provided to those methods which need K specified.

We make comparison by plotting the estimated loading against the truth in different scenarios. To make the results from different methods comparable, we scale the estimated loading and the true loading to unit vector.

In the strong sparsity scenario (Figure 2.1), FLASH shrink all the zero valued points a lot towards zero and keep the larger valued points as they are, which is how the adaptive shrinkage works. In PMD, we use 5-fold OCV (2.4.3) to choose tuning parameters τ_u and τ_v with 10 grids for each, which controls the sparsity level of factor and loading. For SFA, PMD and KSVD, the shrinkage near zero is not big enough, so a lot zero valued points are estimated with big non-zero value. (In this $K = 1$ case, KSVD is same as SVD.)

SFAmix seems to shrink too much on the points near zero. SSVD does a good job in this case, although it seems shrink a bit more to the points near zero. This is not so obvious in sparse case, we can check it in the intermediate sparse case.

In intermediate sparse scenario (Figure 2.2), FLASH, SFA, PMD and KSVD have similar results. In this scenario, we don't expect too much shrinkage on loadings and factors since only 30% of loading values are zero. SSVD and SFAmix shrink too much to the points around zero.

In conclusion, FLASH provides appropriate shrinkage in different scenarios comparing with other methods.

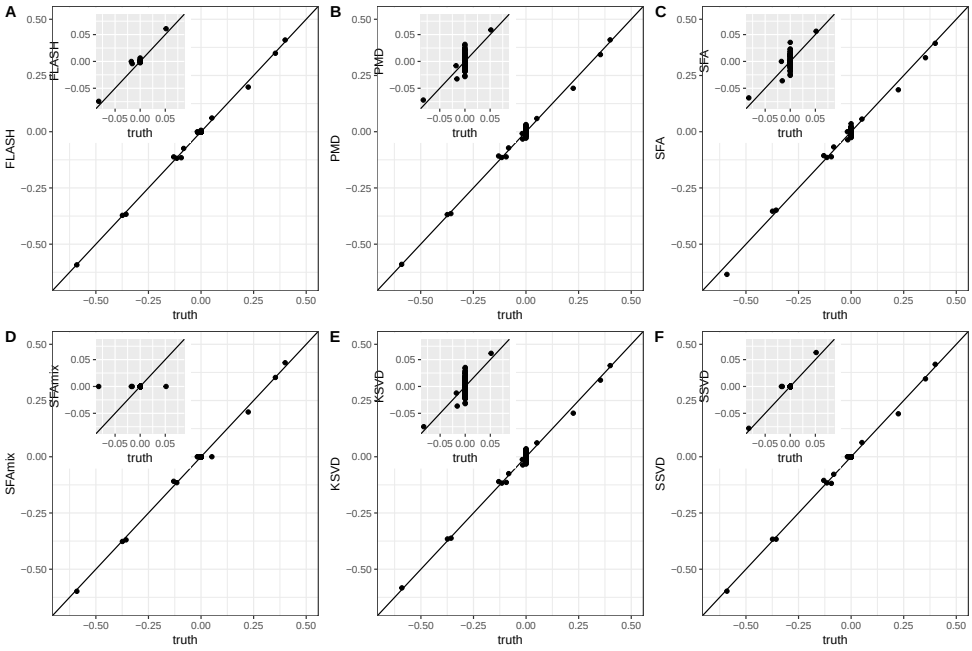

Figure 2.1: This figure shows the comparison on the situation where loading is very sparse. The top left is zoomed plot around zero.

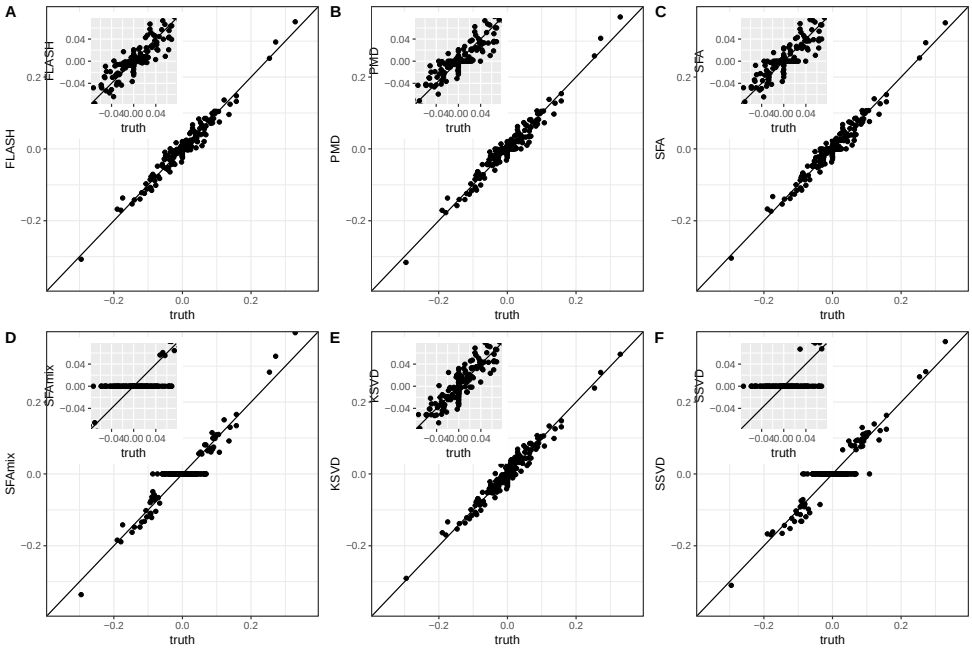

Figure 2.2: This figure shows the comparison on the situation where loading is intermediate sparse. The top left is zoomed plot around zero.

2.4.2 Case $K = 3$

To illustrate the flexibility of FLASH in capturing both sparse and non-sparse factors with simulated data with $K = 3$ factors:

$$
\begin{aligned}
Y &= \boldsymbol{l}_1\boldsymbol{f}_1^T + \boldsymbol{l}_2\boldsymbol{f}_2^T + \boldsymbol{l}_3\boldsymbol{f}_3^T + E && (2.4.5)\\
E_{ij} &= N(0, 1/\tau) && (2.4.6)\\
s_{\boldsymbol{l}_1} = 1, \cdots, 10; &\quad \boldsymbol{l}_{1,s_{\boldsymbol{l}_1}} \sim N(0, 2^2); \boldsymbol{l}_{1,-s_{\boldsymbol{l}_1}} = 0 && (2.4.7)\\
s_{\boldsymbol{l}_2} = 11, \cdots, 60; &\quad \boldsymbol{l}_{2,s_{\boldsymbol{l}_2}} \sim N(0, 1); \boldsymbol{l}_{2,-s_{\boldsymbol{l}_2}} = 0 && (2.4.8)\\
s_{\boldsymbol{l}_3} = 61, \cdots, 150; &\quad \boldsymbol{l}_{3,s_{\boldsymbol{l}_3}} \sim N(0, \frac{1}{2^2}); \boldsymbol{l}_{3,-s_{\boldsymbol{l}_3}} = 0 && (2.4.9)\\
s_{\boldsymbol{f}_1} = 1, \cdots, 80; &\quad \boldsymbol{f}_{1,s_{\boldsymbol{f}_1}} \sim N(0, \frac{1}{2^2}); \boldsymbol{f}_{1,-s_{\boldsymbol{f}_1}} = 0 && (2.4.10)\\
s_{\boldsymbol{f}_2} = 81, \cdots, 160; &\quad \boldsymbol{f}_{2,s_{\boldsymbol{f}_2}} \sim N(0, 1); \boldsymbol{f}_{2,-s_{\boldsymbol{f}_2}} = 0 && (2.4.11)\\
s_{\boldsymbol{f}_3} = 161, \cdots, 240; &\quad \boldsymbol{f}_{3,s_{\boldsymbol{f}_3}} \sim N(0, 2^2); \boldsymbol{f}_{3,-s_{\boldsymbol{f}_3}} = 0 && (2.4.12)
\end{aligned}
$$

where $P = 240$, $N = 150$ and $\tau = 1/4$, and $\boldsymbol{l}_{k,s_{\boldsymbol{l}_k}}$ represent $l_{k,i}$ where $i \in s_{\boldsymbol{l}_k}$; $\boldsymbol{f}_{k,s_{\boldsymbol{f}_k}}$ represent $f_{k,j}$ where $j \in s_{\boldsymbol{f}_k}$, $k = 1, 2, 3$.

This example involves sparsity in both factors and loadings, and the level of sparsity in loadings varies for $k = 1, 2, 3$. This example has a clear bi-cluster structure where each group of samples is loaded only on a subset of variables (Figure 2.3).

We compared a subset of methods (FLASH, SSVD and PMD) that can capture sparsity on both factors and loadings. In PMD, we use 5-fold OCV (2.4.3) to choose the tuning parameter that controls the sparsity level for the factors and loadings. The software does not allow different tuning parameters for each factor and loading.

In SSVD, we used 'ssvd' with default settings and $K = 3$.

Figure 2.3 contains the true underlying structure and estimated structure from different methods. We can see that FLASH has best performance in recovering the real underlying structure. Estimates from PMD and SSVD are either too sparse or too dense.

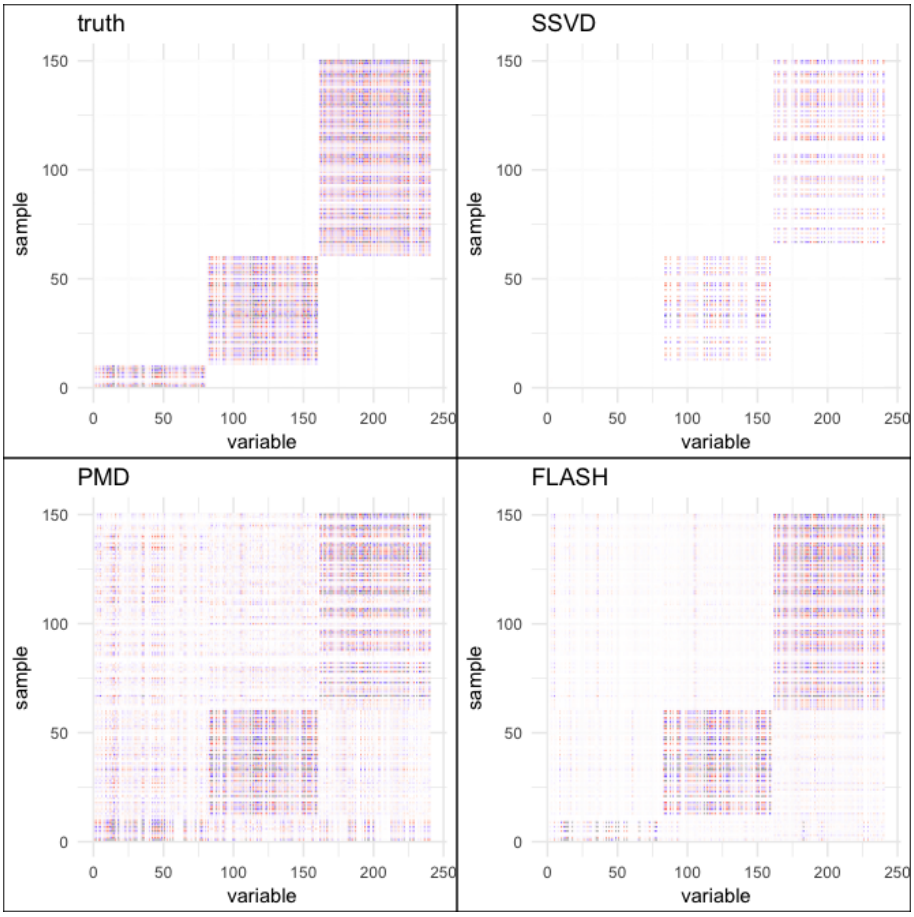

Figure 2.3: Top left: underlying structure; top right: estimated structure from SSVD; bottom left: the estimated structure from PMD, bottom right: the estimated structure of FLASH

2.4.3 Missing data imputation

A common application of factor models is to estimate or "impute" missing elements of a matrix. We now compare several methods on real data by first "hold out" some elements of the data matrix (treating them as missing) and then applying methods to the remaining data to predict the held-out data. We measure performance on held-out data by computing the RMSE:

$$RMSE(\hat{Y}, \Omega) = \sqrt{\frac{1}{|\Omega|} \sum_{ij \in \Omega} (Y_{(ij)} - \hat{Y}_{(ij)})^2}. \tag{2.4.13}$$

where Ω is the set of indices of the held-out data points.

Orthogonal Cross Validation

We introduce a novel approach of cross validation, Orthogonal Cross Validation (OCV), to choose the "hold-out" pattern and randomly divide the data matrix into submatrices. OCV is a k-fold CV by choosing orthogonal parts from data matrix in each fold. The details of k-fold CV is provided in Algorithm 2.

Algorithm 2 k-fold CV

procedure K-FOLD CROSS VALIDATION

randomly divide Y into $Y_{(1)}, \cdots, Y_{(k)}$ with "hold-out" index $\Omega_{(1)}, \cdots, \Omega_{(k)}$

for $i = 1, \cdots, k$ **do**

take $Y_{(i)}$ as missing and run FLASH

$\hat{Y}_{(i)} = E[Y_{\Omega_{(i)}} | Y_{-\Omega_{(i)}}]$

$s_i^2 = ||\hat{Y}_{(i)} - Y_{\Omega_{(i)}}||_2^2$

return RMSE: $score = \sqrt{\frac{\sum_k s_k^2}{NP}}$

Here, we choose RMSE (2.4.13) as the score function. To choose the "hold-out" pattern, we randomly divide the columns into k sets and the rows into k sets as well, and put these sets into k orthogonal parts, and then take all Y_{ij} with the chosen column and row indices as "hold-out" $Y_{(i)}$.

To illustrate this scheme, we take 3-fold CV as an example. We randomly divide the the columns into 3 sets and the rows into 3 sets as well. The data matrix Y is divided into 9 partition (by row and column permutation):

$$Y = \begin{pmatrix} Y_{11} & Y_{12} & Y_{13} \\ Y_{21} & Y_{22} & Y_{23} \\ Y_{31} & Y_{32} & Y_{33} \end{pmatrix}.$$

Then $Y_{(1)} = \{Y_{11}, Y_{22}, Y_{33}\}$, $Y_{(2)} = \{Y_{12}, Y_{23}, Y_{31}\}$ and $Y_{(3)} = \{Y_{13}, Y_{21}, Y_{32}\}$ are

orthogonal to each other. Then the data matrix Y is marked as:

$$Y = \begin{pmatrix} Y_{(1)} & Y_{(2)} & Y_{(3)} \\ Y_{(3)} & Y_{(1)} & Y_{(2)} \\ Y_{(2)} & Y_{(3)} & Y_{(1)} \end{pmatrix}.$$

In OCV scheme, we take factor analysis as an example, which is based on the model with low rank structure: LF^T. In each fold k, $Y_{(k)}$ contains equally balanced part of data matrix and includes all the row and column indices. This also helps to make sure that all i's and j's are included into each $Y_{-(k)}$. Otherwise, we can't make prediction when whole columns or rows are missing. In 3-fold OCV, we have:

$$Y = \begin{pmatrix} Y_{11} & Y_{12} & Y_{13} \\ Y_{21} & Y_{22} & Y_{23} \\ Y_{31} & Y_{32} & Y_{33} \end{pmatrix} = \begin{bmatrix} L^{(1)} \\ L^{(2)} \\ L^{(3)} \end{bmatrix} \times \begin{bmatrix} F^{(1)} & F^{(2)} & F^{(3)} \end{bmatrix} + E \tag{2.4.14}$$

$$= \begin{pmatrix} Y_{(1)} & Y_{(2)} & Y_{(3)} \\ Y_{(3)} & Y_{(1)} & Y_{(2)} \\ Y_{(2)} & Y_{(3)} & Y_{(1)} \end{pmatrix} = \begin{bmatrix} L^{(1)}F^{(1)} & L^{(1)}F^{(2)} & L^{(1)}F^{(3)} \\ L^{(2)}F^{(1)} & L^{(2)}F^{(2)} & L^{(2)}F^{(3)} \\ L^{(3)}F^{(1)} & L^{(3)}F^{(2)} & L^{(3)}F^{(3)} \end{bmatrix} + E \tag{2.4.15}$$

where $Y_{(1)} = \{Y_{11}, Y_{22}, Y_{33}\}$, $Y_{(2)} = \{Y_{12}, Y_{23}, Y_{31}\}$ and $Y_{(3)} = \{Y_{13}, Y_{21}, Y_{32}\}$. We can see for each "hold-out" part, $Y_{(k)}$, $L^{(1)}, L^{(2)}, L^{(3)}$ and $F^{(1)}, F^{(2)}, F^{(3)}$ show up once and only once. It means that in each fold, the "hold-out" part contains equally all the row information and all the column information. This balanced "hold-out" pattern is also useful to find estimation of rank K in other dimensional reduction methods to provide stable estimation.

To measure performance we use the Root Mean Square Error (RMSE) (2.4.13). Here we define the index of the missing values as a set Ω. We use square root of F-norm between imputed values and the true values in the "hold-out" position. In this OCV approach, every data point is held out once and only once, so Ω is all the indices. And we calculate the mean of square error for all data points and take square root to MSE.

We applied FLASH, NSFA, PMD, and softImpute (Mazumder et al., 2010) to 5 data sets. We apply 10-fold OCV (Orthogonal Cross Validation, 2.4.3) on the observed data to decide the "hold-out" pattern and treat the "hold-out" data points as missing each time. To assess the missing value predictions from different methods we use RMSE (2.4.13) as criteria. We repeat this procedure 100 times for each data set.

Movie data

The MovieLens data set (Harper and Konstan, 2016) is available at the GroupLens Research group which is usually used for the exploration of recommendation systems. There are several data sets available for stable bench mark and we choose the smallest one. The MovieLens 100K data set we used here is a data set with about 100K ratings from around 1000 users and 1700 movies. The ratings are integers from 1 to 5 representing the preference of the users. In this data, all the ratings are nonnegative, and we center and scale the data for each user. We focus on the personal preference of each user on movies, so the pre-processed data for each user would still represent his/her preference as the same. There are 94% data are unobserved, and we only

hold out part of the observed data.

GTEx eQTL summary statistic

This data set is summary statistics (Z-score) from eQTL analysis. The size of this data matrix is 16,069 by 44, where the rows index different SNP-gene pairs and the columns index tissues (or cell types). We used the Z-scores for candidate local ("cis") eQTLs for each gene. More details of this data set are described in 2.4.4.

Brain Tumor data

This data set is a microarray data containing 43 brain tumors of four types and 356 continous variable related with the expression data, which is from 'denoiseR' package (Josse et al., 2016). The methods are directly applied on the data matrix without any pre-processing. We set the rank equal to 20 for PMD and softImpute as input, since we find in practice that the rank of the estimated low rank structure is almost always smaller than 20, which is around 15-20.

Presidential address data

This data set is a contingency table cross-tabulating with 13 rows and 836 columns, where rows stand for US presidents from 1940 to 2009 and columns correspond to the words they used in their inaugural addresses, which comes from 'denoiseR' (Josse et al., 2016) as well. We pre-process the data by centering and scaling on both rows

and columns to transform the data matrix as follows:

$$\frac{Y_{ij} - \alpha_i - \beta_j}{\gamma_i t_j} \tag{2.4.16}$$

which makes the transformed data have zero row and column means and unit row and column variances. This 'biScale' pre-process is implemented in 'softImpute'. We apply 'biScale' to this data because not only the frequency of words varies among different presidents, but also the frequency of each word varies a lot. This data is different from movie rating data of which the ratings can only take values from 0 to 5. Some words show up more frequently than others, such as "a" and "the", so we should center and scale the data by columns (word) as well as rows (samples).

Breast cancer data

The Breast cancer data is used in (Knowles and Ghahramani, 2011) as an example, with 251 samples and 226 genes, which is originally from (Carvalho et al., 2008). We set the rank equal to 50 which is close the rank of the NSFA results. Based on our experience, the rank estimated by FLASH is also around 40-50. Following the pre-processing procedure from NSFA, we also use the centered data.

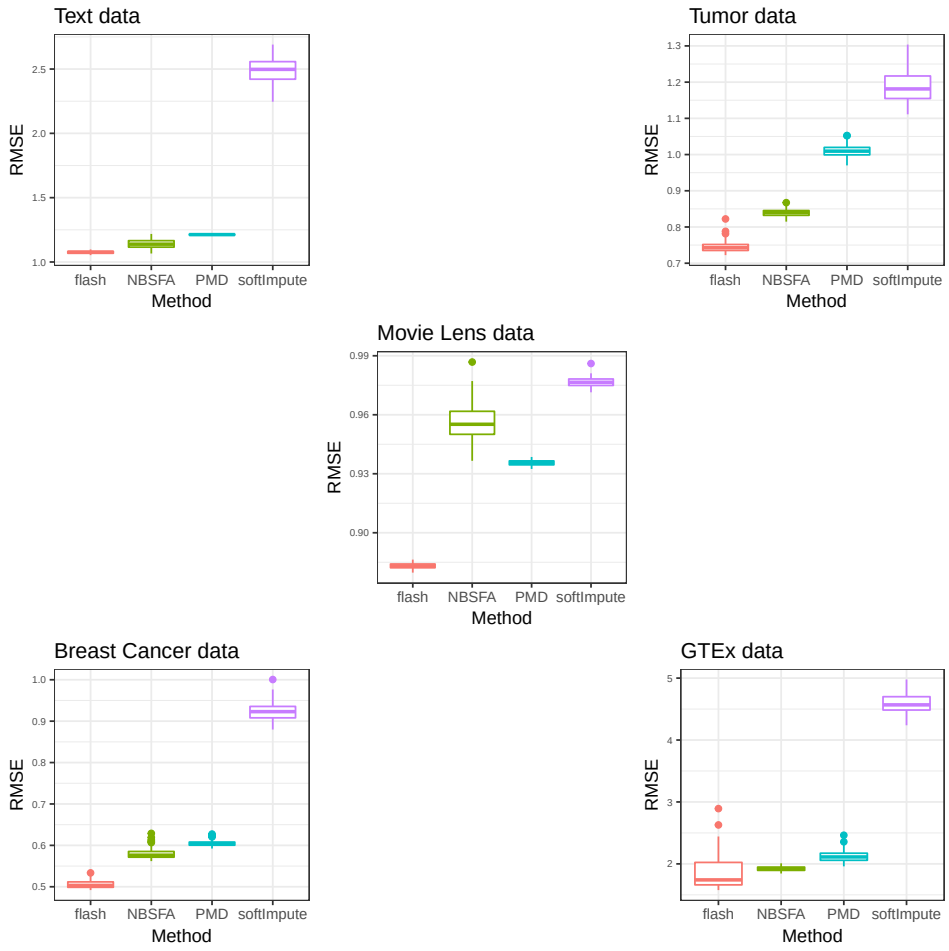

Figure 2.4: This figure shows the comparison of RMSE in missing value imputation from different methods

For each data set, we use 100 simulations with different random seeds to choose the "hold out" pattern. The RMSE of "hold out" data prediction is calculated based on 2.4.13. More details about "hold out" pattern of OCV and the way to get RMSE are included in 2.4.3. Figure 2.4 shows the RMSE from different methods among 100 simulations in these 5 data sets. We can see that FLASH outperforms

other imputation methods with respect to precision on prediction of held-out data (missing value imputation).

2.4.4 Sharing of genetic effects on gene expression among tissues

We now apply FLASH to assess sharing of genetic effects on gene expression across human tissues. Specifically we apply FLASH to data from the Genotype Tissue Expression (GTEx) project (Consortium et al., 2015b), which measures the effects of thousands of "eQTLs" in dozens of human tissues. (An eQTL is a genetic variant that is associated with expression of a gene.) To identify eQTLs, in each of 44 human tissues, at each of $16,069$ genes, the project tested for association between expression and every near-by genetic variant, each test yielding a Z score. Here we first identify the most significant genetic variant for each gene (the "top" eQTL), and form a matrix of the Z scores for these top eQTLs.

Thus the input to FLASH is a $16,069 \times 44$ matrix of Z scores, with Z_{ij} reflecting the strength (and direction) of effect of eQTL j in tissue i. We applied FLASH with the greedy approach, followed by backfitting, which yielded 26 factors.

The inferred factors (Figure 2.5-2.6; Table 2.1) each of which is a vector of length 44, provide a helpful summary of the main patterns of eQTL sharing among tissues (and, conversely, the main patterns of tissue-specificity). For example, the first factor has almost equal weight for every tissue, and reflects the fact that many eQTLs show similar effects across all 44 tissues. The second factor has strong effects only in the 10 brain tissues, from which we infer that some eQTLs show much stronger (or weaker) effects in brain tissues than other tissues.

Subsequent factors tend to be sparser, and many have a strong effect in only one tissue, capturing "tissue-specific" effects. For example, the 3rd factor shows a strong effect only in whole blood, and captures eQTLs that have much stronger (or weaker) effects in whole blood than other tissues. (Two tissues, "Lung" and "Spleen", show very small effects in this factor but with the same sign as blood. This is intriguing since the lung has recently been found to make blood cells (Lefrançais et al., 2017) and a key role of the spleen is storing of blood cells.) Similarly Factors 7, 11 and 14 capture effects specific to "Testis", "Thyroid" and "Esophagus Mucosa".

A few other factors show strong effects in a small number of tissues that are known to be biologically related, providing external scientific support for the results. For example, factor 10 captures the two tissues related to the cerebellum, "Brain Cerebellar Hemisphere" and "Brain Cerebellum". Factor 19 captures tissues related to female reproduction, "Ovary", "Uterus" and "Vagina". Factor 5 captures "Muscle Skeletal", with small but concordant effects in the heart tissues ("Heart Atrial Appendage" and "Heart Left Ventricle"). Factor 4, captures the two skin tissues ("Skin Not Sun Exposed Suprapubic", "Skin Sun Exposed Lower leg") and also "Esophagus Mucosa", possibly reflecting the sharing of squamous cells that are found in both the surface of the skin, and the lining of the digestive tract. In factor 24, "Colon Transverse" and "Small Intestine Terminal Ileum" show the strongest effects (and with same sign), reflecting some sharing of effects in these intestinal tissues.

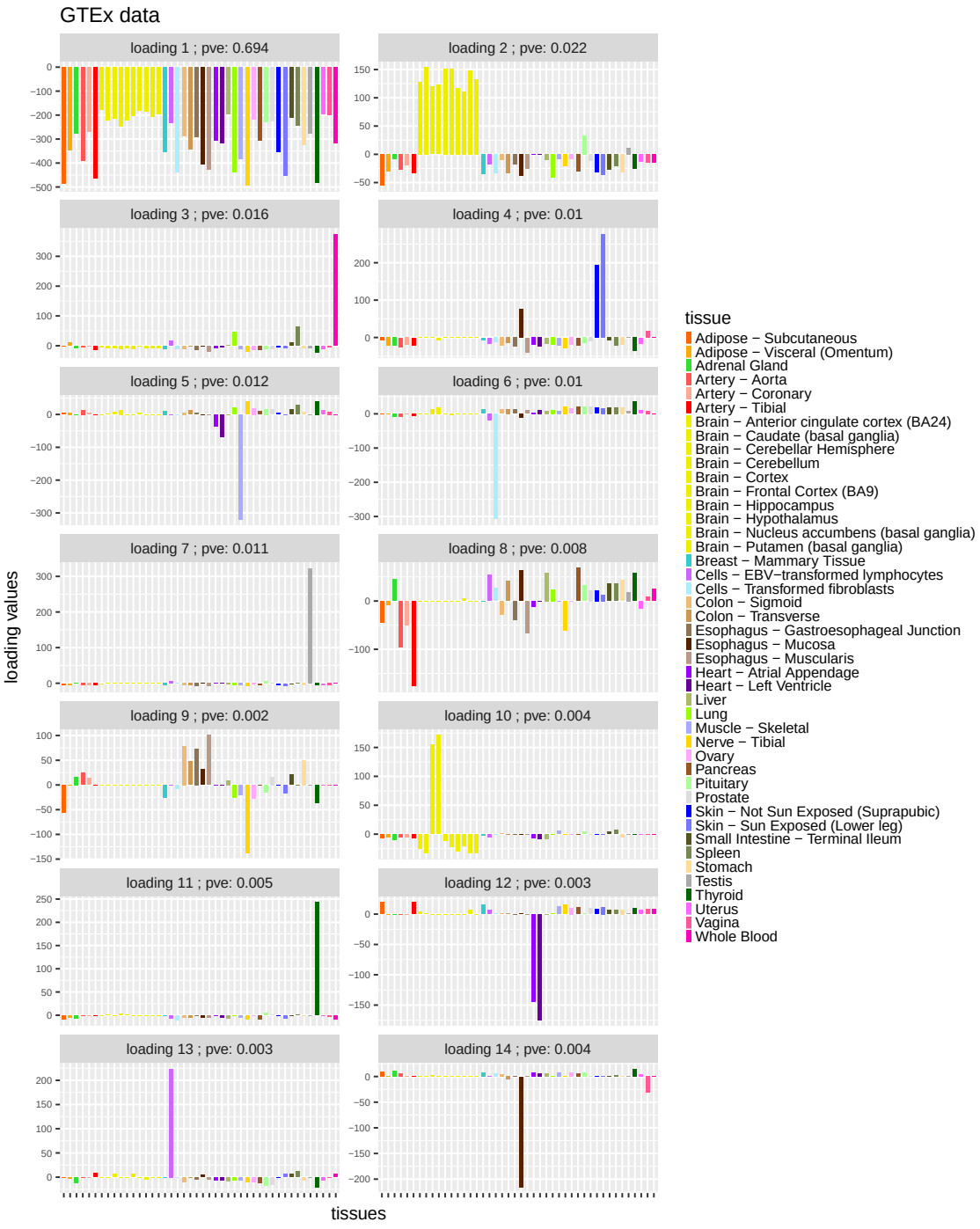

Figure 2.5: Factor 1 - Factor 14

Figure 2.6: Factor 15 - Factor 26

2.4.5 Application on video data

In this part, we apply FLASH on the video data sets. Each video data is a series of pictures. We convert the matrix of each picture to a vector, and we put the vectors together as rows into a data matrix in chronological order. In this way, we get the data matrix for each video. We use two data sets as examples to show the application of FLASH in denoising video data and separating background and foreground.

Waving tree

This data is from https://www.microsoft.com/en-us/research/project/test-images-for-wallflower-paper. It is rgb data, and we only take the second panel of each picture. We apply FLASH with maximum rank as 30 after transforming the video data into matrix as mentioned above. Figure 2.7-2.8 show the result of rank 30 approximation from FLASH. FLASH decomposes the data matrix into three parts. The first part is from factor 1, which represents the background. Factor 2-3 construct the second part, which captures the periodically moving item, the waving tree. The rest factors represent the foreground which shows up in a short period of time. There is no noise in this data set, so we should use full rank decomposition from FLASH. However, from the residual picture and rank 30 approximation picture in the figures, we can see that this rank 30 approximation contains most information from the video. The whole decomposed video is provided online (https://drive.google.com/file/d/0B5EW2zUIpvHpWC14b2lxTm1CVFE/view?usp=sharing).

Table 2.1: Summary of tissues contributing most strongly to each tissue

Factor	*Corresponding Tissues*
1	All tissues
2	"Brain Anterior cingulate cortex BA24", "Brain Caudate basal ganglia", "Brain Cerebellar Hemisphere, "Brain Cerebellum", "Brain Cor- tex", "Brain Frontal Cortex BA9", "Brain Hippocampus", "Brain Hypothalamus", "Brain Nucleus accumbens basal ganglia", "Brain Putamen basal ganglia"
3	"Whole Blood"
4	"Skin Not Sun Exposed Suprapubic", "Skin Sun Exposed Lower leg", "Esophagus Mucosa"
5	"Muscle Skeletal", "Heart Atrial Appendage", "Heart Left Ventricle"
6	"Cells Transformed fibroblasts"
7	"Testis"
8	"Artery Aorta", "Artery Coronary", "Artery Tibial" "Nerve Tibial", "Adipose Subcutaneous", "Adrenal Gland","Colon Transverse", "Cells EBV-transformed lymphocytes","Esophagus Mucosa", "Esophagus Muscularis","Liver", "Pancreas", "Stomach", "Thyroid"
9	"Adipose Subcutaneous", "Colon Sigmoid", "Colon Transverse", "Stomach" "Esophagus Gastroesophageal Junction", "Nerve Tibial", "Esophagus Mus-cularis"
10	"Brain Cerebellar Hemisphere" and "Brain Cerebellum"
11	"Thyroid"
12	"Heart Atrial Appendage" and "Heart Left Ventricle"
13	"Cells EBV-transformed lymphocytes"
14	"Esophagus Mucosa"
15	"Artery Aorta", "Artery Coronary", "Artery Tibial", "Esophagus Muscularis", "Nerve Tibial"
16	"Nerve Tibial"
17	"Adipose Subcutaneous", "Adipose Visceral Omentum", "Breast Mammary Tissue"
18	"Pancreas"
19	"Ovary", "Uterus", "Vagina"
20	"Colon Sigmoid", "Colon Transverse", "Stomach", "Esophagus Gastroesophageal Junction", "Esophagus Muscularis"
21	"Lung"
22	"Adrenal Gland"
23	"Liver"
24	"Colon Transverse" and "Small Intestine Terminal Ileum"
25	"Pituitary"
26	"Spleen"

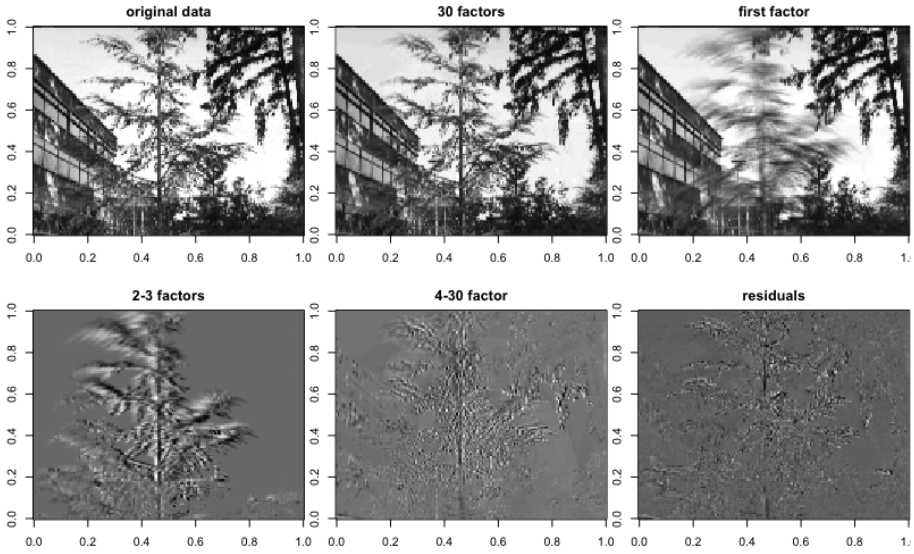

Figure 2.7: Decomposition result from FLASH for waving tree data.

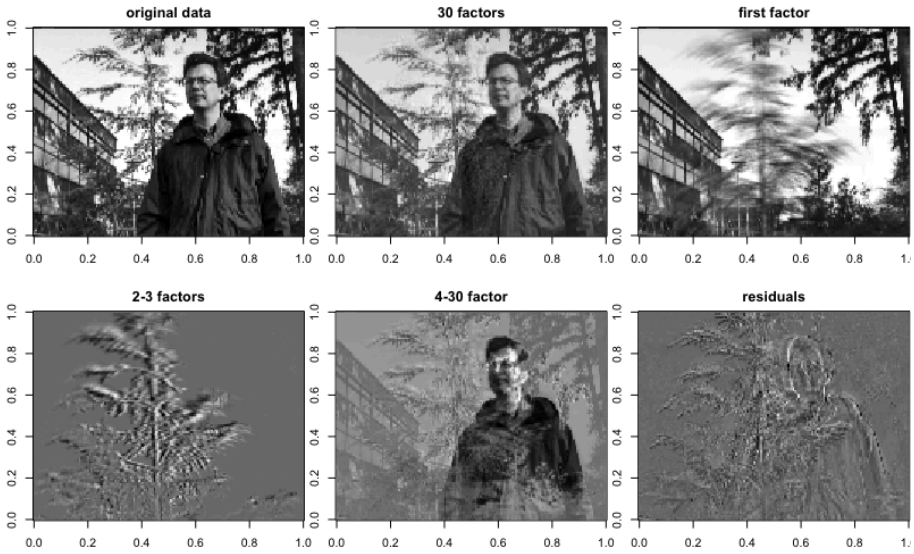

Figure 2.8: Decomposition result from FLASH for waving tree data.

PINCAT numerical phantom

This video data is from https://statweb.stanford.edu/ candes/SURE/example1.html (Candès et al., 2013; Sharif and Bresler, 2007). In this data set, each element is a complex number, and we take the modulus of these complex numbers to get the data matrix of real numbers. We add artificial noise to the data matrix, where the noise term follows $i.i.d\ N(0, 30)$. We apply FLASH with greedy algorithm to get a rank 16 approximation. From the results in Figure 2.9-2.10, we can see that our method performs very well by separating signal and noise from the noisy data. There is almost no signal left in the residual matrix. The result of the whole video is provided online (https://drive.google.com/file/d/0B5EW2zUIpvHpMTNSc3pxczhrdXc/view?usp=sharing).

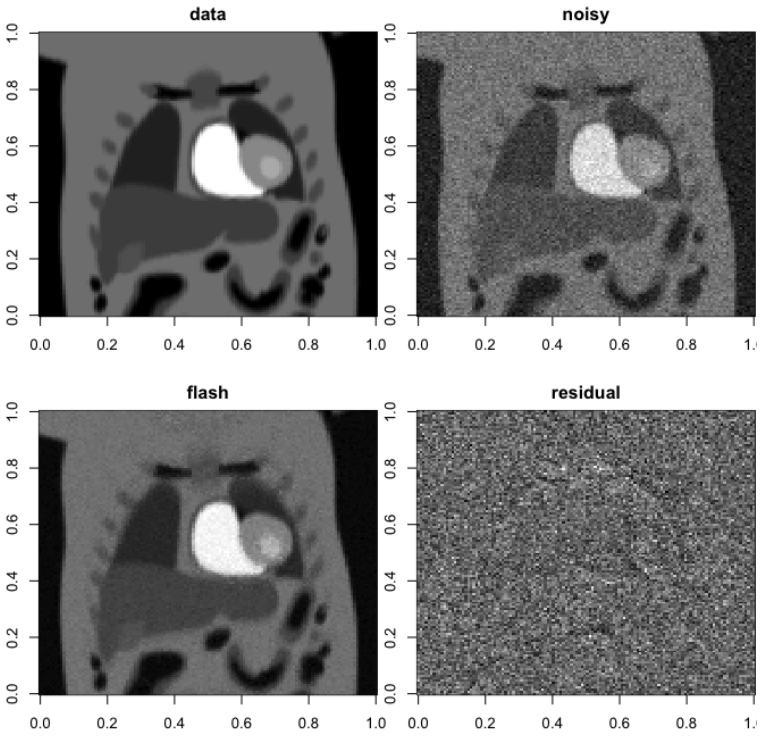

Figure 2.9: Denoising video data by FLASH.

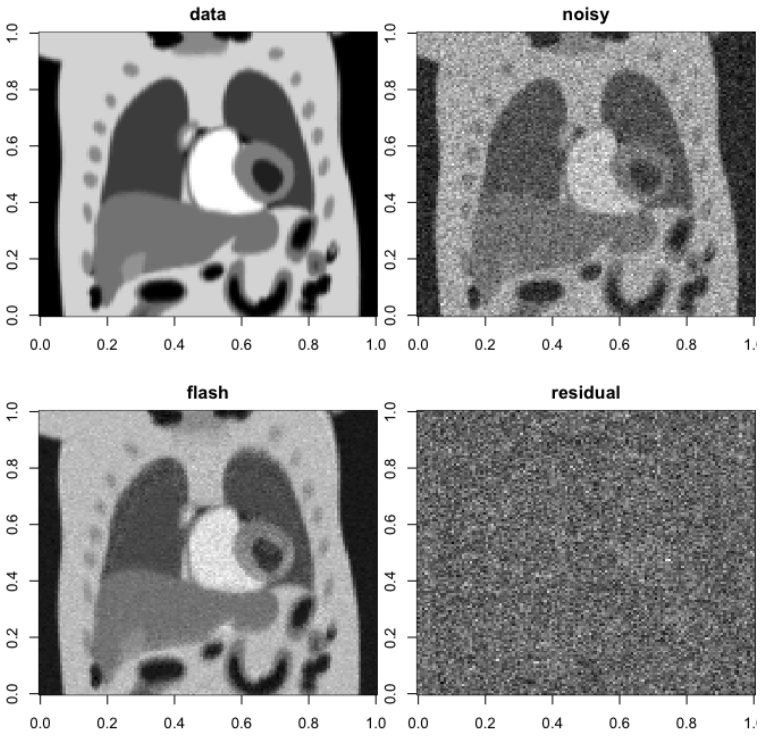

Figure 2.10: Denoising video data by FLASH.

2.5 Discussion

In this article, we introduce a general Bayesian Sparse factor model using adaptive shrinkage prior on both factors and loadings. We describe its application in the context of factor analysis on summary statistics of eQTL analysis of GTEx data to find the underlying structure. We find that adaptive shrinkage prior inducing flexible sparsity helps interpretation in factor loadings, which provides biologically meaningful results. Adaptive shrinkage prior provides advanced properties to approximate any symmetric unimodal density. For factor analysis or matrix factorization, there

are several interesting directions for this research and easy extensions of FLASH in this framework.

One of the interesting extensions is sparse solution of non-negative matrix factorization (NMF), which is a special case when factors and loadings are non-negative and the noise is Gaussian white noise with variance equal to zero. In "ashr" package, there is an option that we can choose the prior to be half uniform or half normal, which leads to the final estimate as non-negative and provides proper adaptive shrinkage. With this option, we can apply FLASH to non-negative matrix factorization with proper initial values. For example, we can use NMF result as the input of initial value in FLASH backfitting algorithm to get non-negative and flexible sparse solutions of factors and loadings.

In practical applications of factor analysis, there might be some known covariates, which can be considered as known factors (fixed factors). Given the factors are known, we only need to do inference on the corresponding loadings. Our FLASH framework also allows some of the factors to be known. The details of the algorithm is included in the Appendix A.7.

In FLASH framework, it is also easy to extend our method to tensor factorization. We mainly focus on the situations with data matrix as input in FLASH algorithm, while the factor model can be also applied to tensor data (we take 3-mode array as example) as follows:

$$Y_{ijm} = \sum_k l_i^{(k)} f_j^{(k)} h_m^{(k)} + E_{ijm} \tag{2.5.1}$$

where E_{ijm} are iid Gaussian white noise and $l_i^{(k)} \sim g_{l^{(k)}}(\cdot)$, $fl_j^{(k)} \sim g_{f^{(k)}}(\cdot)$, $h_m^{(k)} \sim g_{h^{(k)}}(\cdot)$ and $g_{h^{(k)}}(\cdot)$, $g_{l^{(k)}}(\cdot)$ and $g_{f^{(k)}}(\cdot)$ are ASH prior. To apply FLASH algorithm, we can apply EBNM approach consecutively on each mode, l, f and h, in rank one case. In multiple rank scenario, both greedy algorithm and backfitting algorithm are applicable here. One of the computational advantages of FLASH in tensor factor analysis is that we don't need to choose the tuning parameters in each mode. If we extend PMD algorithm in tensor decomposition, we need to choose the tuning parameters using cross-validation of each mode where the computational complexity increases exponentially as the number of mode becomes larger.

Choosing the number of latent factors of the low-rank structure is a hard problem. We can add one more extremely sparse rank one matrix with tiny non-zero values to existing estimated low-rank structure which would not affect prediction or estimation. There is no theoretical result about how precise the rank is. In practice, we make decisions on rank based on the estimation of the factors and loadings. In greedy algorithm (Appendix A.4), we stop at the number of rank when newly added factor or corresponding loading are shrunken to zero. In backfitting algorithm (Appendix A.4), after deleting the factors which are shrunken to zero, the estimated rank is the number of remaining factors.

CHAPTER 3

LOGISTIC FLASH

3.1 Introduction

In this chapter, we extend the FLASH method in chapter 2 to discrete data. FLASH model is based on normal distribution and each entry of data matrix takes value in real numbers. In this chapter, we consider factor analysis with data matrix in which the entries are discrete rather than real-valued. Gaussian distribution is canonical distribution with appealing theoretical properties, but it might be inappropriate if the data is discrete, nonnegative. Therefore we consider Binomial distribution, Bernoulli distribution and Poisson distribution (compound Poisson distribution) for discrete data. In practice, discrete data is common in many areas including education, political science, ecology, computational biology, recommendation system and social network. For example, the binary element can represent yes/no, positive/negative, like/dislike and true/false in survey, recommendation system and other data sets (Davenport et al., 2014; Lan et al., 2014). Binomial data is also of interests, such as observed genotypes in the presence of complex population structure in genetics (Hao et al., 2016).

Logistic factor analysis generalizes factor analysis in the same way as regression is generalized by logistic regression. We assume that the binary data follows Bernoulli distribution according to log-odds that is parameterized by the corresponding element in low rank matrix, which is a linear combination of latent factors. There exist many possible link functions for binary data. Logistic function is one of the canoni-

cal link functions to parameterize the probability parameter in Bernoulli distribution (McCullagh and Nelder, 1989).

To estimate or recover the underlying low rank structure given the binary data matrix, there are many methods proposed such as binary PCA, logistic PCA and 1-bit matrix completion (Davenport et al., 2014; Bhaskar and Javanmard, 2015; Cai and Zhou, 2013; Collins et al., 2001; de Leeuw, 2006; Schein et al., 2003). Sparse solution of the problem is also investigated including (Lee et al., 2010; Lan et al., 2014). Sparsity in loadings and factors can enhance interpretation of results to alleviate the identifiability issue in factor analysis and improve the accuracy of prediction by keeping representative elements in factors and loadings.

We apply adaptive shrinkage prior (Stephens, 2016) as sparse-introducing prior for both factors and loadings and develop an iterative procedure based on variational EM algorithm for inference in logistic factor analysis. In practice, it's hard to know the sparse level of factors and loadings since the latent structure is unobserved. Adaptive shrinkage prior is a flexible and data-driven prior to capture the prior belief of the underlying structure.

In variational EM algorithm, we find that maximizing the lower bound (evidence lower bound) of log likelihood of the data is difficult, since it hard to find closed form for the updates of parameters in this optimization with the Bernoulli or Binomial likelihood. We use a tight lower bound as the objective function based on convexity inequality introduced in Jaakkola and Jordan (2000). We can apply an Empirical Bayes (EB) approach for normal means problem to solve the optimization problem of the tight lower bound. For binomial data, we apply variational inference with

Polya-Gamma augmentation (Klami, 2014; Polson et al., 2013), which also ends up solving EB approach for normal means problem in 1.0.1. As binary model is a special case of binomial model, we find that these two different procedures on binary data have same updates in each iteration.

3.2 Methods

3.2.1 Logistic FLASH model (rank one)

Y denotes the observed $N \times P$ data matrix with each element, Y_{ij} with $i = 1, \cdots, N$ and $j = 1, \cdots, P$, taking values in ± 1. Logistic FLASH model is as follows:

$$Y_{ij} = 2 * Bern(p_{ij}) - 1 \tag{3.2.1}$$

$$p_{ij} = p(Y_{ij} = 1 | Z_{ij}) \tag{3.2.2}$$

$$\log \frac{p_{ij}}{1 - p_{ij}} = Z_{ij} \tag{3.2.3}$$

$$Z_{ij} = l_i f_j \tag{3.2.4}$$

where we assume that

$$l_i \sim g_{\boldsymbol{l}}(\cdot) \tag{3.2.5}$$

$$f_j \sim g_{\boldsymbol{f}}(\cdot). \tag{3.2.6}$$

$g_{\boldsymbol{l}}$ and $g_{\boldsymbol{f}}$ are from some distribution family, such as $\mathcal{U}$, $\mathcal{S}U$ and $\mathcal{S}N$.

Variational Inference for rank one model

Our goal is to estimate both distributions $g_{\boldsymbol{l}}, g_{\boldsymbol{f}}$ and the posterior of $\boldsymbol{l}, \boldsymbol{f}$. Following the same idea in chapter 2, we introduce variational EM algorithm to maximize ELBO, $F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}})$, of the log-likelihood.

$$F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}) = \iint q_{\boldsymbol{f}}(\boldsymbol{f}) q_{\boldsymbol{l}}(\boldsymbol{l}) \log \frac{p(Y|\boldsymbol{l}, \boldsymbol{f}) p(\boldsymbol{l}|g_{\boldsymbol{l}}) p(\boldsymbol{f}|g_{\boldsymbol{f}})}{q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f})} d\boldsymbol{f} d\boldsymbol{l}.$$

Based on Jaakkola and Jordan (2000) we can find a tight lower bound for the ELBO. Since $\log \frac{P(Y_{ij}=1|Z_{ij})}{P(Y_{ij}=-1|Z_{ij})} = Z_{ij}$ and $Y_{ij} = \pm 1$, we can write

$$P(Y_{ij}|\boldsymbol{l}, \boldsymbol{f}) = P(Y_{ij}|Z_{ij}) = h(Y_{ij} Z_{ij}) = \frac{1}{1 + exp(-Y_{ij} Z_{ij})} \tag{3.2.7}$$

where $h(z)$ has a tight lower bound with parameter ξ_z:

$$\begin{aligned} h(z) &\geq h(\xi_z) exp(\frac{z - \xi_z}{2} - \tau(\xi_z)(z^2 - \xi_z^2)) && (3.2.8) \\ \tau(\xi_z) &= \frac{1}{2\xi_z}(h(\xi_z) - \frac{1}{2}) && (3.2.9) \\ &= \frac{1}{4\xi_z} \tanh(\frac{\xi_z}{2}). \end{aligned}$$

We denote the tight lower bound of $F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}})$ as $Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \xi)$:

$$\begin{aligned} F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}) &\geq \iint q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f}) \log \frac{p(\boldsymbol{l}|g_{\boldsymbol{l}}) p(\boldsymbol{f}|g_{\boldsymbol{f}}) H(\boldsymbol{l}, \boldsymbol{f}, Y, \xi)}{q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f})} d\boldsymbol{l} d\boldsymbol{f} \qquad (3.2.10) \\ &\equiv Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \xi) \end{aligned}$$

where

$$H(\boldsymbol{l}, \boldsymbol{f}, Y, \xi) = \exp\left\{ \sum_{ij} [\frac{Y_{ij} l_i f_j - \xi_{ij}}{2} + \log(h(\xi_{ij})) - \tau(\xi_{ij})(l_i^2 f_j^2 - \xi_{ij}^2)] \right\} \quad (3.2.11)$$

$$\tau(\xi) = \frac{1}{4\xi} \tanh(\frac{\xi}{2}). \quad (3.2.12)$$

$(\xi_{N\times P})_{(ij)} = \xi_{ij}, i = 1, \cdots, N, j = 1, \cdots, P$ are variational parameters.

Since $Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \xi)$ is a tight lower bound of $F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}})$,

$$\max F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}) = \max Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \xi). \quad (3.2.13)$$

The procedure we propose is an alternating maximization, which alternates between optimizing $q_{\boldsymbol{l}}, g_{\boldsymbol{l}}$ and $q_{\boldsymbol{f}}, g_{\boldsymbol{f}}$. And then, we maximize the lower bound over parameters ξ. By plugging in the formula of $H(\boldsymbol{l}, \boldsymbol{f}, Y, \xi)$ into $Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \xi)$, we can get the explicit form as follows:

$$\begin{aligned} Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \xi) &= E_q \sum_{ij} \left\{ \frac{Y_{ij} l_i f_j - \xi_{ij}}{2} + \log(h(\xi_{ij})) - \tau(\xi_{ij})(l_i^2 f_j^2 - \xi_{ij}^2) \right\} \\ &\quad + E_q \log p(\boldsymbol{l}|g_{\boldsymbol{l}}) p(\boldsymbol{f}|g_{\boldsymbol{f}}) - E_q \log q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f}). \end{aligned}$$

Based on Lemma 1 in Appendix A.1, the optimization of $Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \xi)$ over $q_{\boldsymbol{l}}, g_{\boldsymbol{l}}$

(given $q_{\boldsymbol{f}}, g_{\boldsymbol{f}}, \xi$) can be solved by EB approach of normal means problem with:

$$x_i = \frac{\frac{1}{2}\sum_j (Y_{ij} E f_j)}{2\sum_j (\tau(\xi_{ij}) E f_j^2)} \tag{3.2.14}$$

$$s_i^2 = \frac{1}{2\sum_j (\tau(\xi_{ij}) E f_j^2)}. \tag{3.2.15}$$

Similarly, we can apply EB approach of normal means problem to solve the optimization over $q_{\boldsymbol{f}}, g_{\boldsymbol{f}}$ (given $q_{\boldsymbol{l}}, g_{\boldsymbol{l}}, \xi$) with normal means parameters as follows:

$$x_j = \frac{\frac{1}{2}\sum_i (Y_{ij} E l_i)}{2\sum_j (\tau(\xi_{ij}) E l_i^2)} \tag{3.2.16}$$

$$s_j^2 = \frac{1}{2\sum_i (\tau(\xi_{ij}) E l_i^2)}. \tag{3.2.17}$$

Then, we need to maximize over ξ to make this lower bound tight given current $g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, q_{\boldsymbol{l}}, q_{\boldsymbol{f}}$. From (3.2.8), we can see that ξ_z equal to z maximized the tight lower bound. So the update of ξ_{ij} in our variational algorithm would be:

$$\xi_{ij}^2 = E_q Z_{ij}^2 = E_q l_i^2 E_q f_j^2. \tag{3.2.18}$$

More details of the variational algorithm are included in Algorithm 3.

Algorithm 3 Logistic FLASH rank one

1: **procedure** VARIATIONAL EM ALGORITHM

2: $t = 0$

3: initialize $q_{\boldsymbol{l}}^0$, $q_{\boldsymbol{f}}^0$, $g_{\boldsymbol{l}}^{(0)}$, $g_{\boldsymbol{f}}^{(0)}$ and $\xi_{(0)}^2$

4: $\epsilon = 1$

5: **while** $\epsilon > 10^{-5}$ **do**

6: $t = t + 1$

7: $q_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{l}}^{(t)} \leftarrow \underset{q_{\boldsymbol{l}}, g_{\boldsymbol{l}}}{\arg\max} Q(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}^{(t-1)}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}^{(t-1)}, \xi_{(t-1)}^2)$

8: $q_{\boldsymbol{f}}^{(t)}, g_{\boldsymbol{f}}^{(t)} \leftarrow \underset{q_{\boldsymbol{f}}, g_{\boldsymbol{f}}}{\arg\max} Q(q_{\boldsymbol{l}}^{(t)}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{f}}, \xi_{(t-1)}^2)$

9: $\xi_{(t)}^2 \leftarrow \underset{\xi^2}{\arg\max} Q(q_{\boldsymbol{l}}^{(t)}, q_{\boldsymbol{f}}^{(t)}, g_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{f}}^{(t)}, \xi^2)$

10: $\epsilon = \left| Q(q_{\boldsymbol{l}}^{(t)}, q_{\boldsymbol{f}}^{(t)}, g_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{f}}^{(t)}, \xi_{(t)}^2) - Q(q_{\boldsymbol{l}}^{(t-1)}, q_{\boldsymbol{f}}^{(t-1)}, g_{\boldsymbol{l}}^{(t-1)}, g_{\boldsymbol{f}}^{(t-1)}, \xi_{(t-1)}^2) \right|$

return $list(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}})$

In step 7 and 8 in Algorithm 3, we apply *ash* method as EB approach for normal means problem to solve the optimizations in those steps with updates (3.2.14)-(3.2.17). In step 9, the update of ξ follows (3.2.18) for each ξ_{ij}, $i = 1, \cdots, N$ and $j = 1, \cdots, P$.

3.2.2 Rank K model

In this section, we consider a model with multiple factors:

$$\begin{aligned} Y_{ij} &= 2 * Bern(p_{ij}) - 1 \\ \log \frac{p_{ij}}{1 - p_{ij}} &= Z_{ij} \\ Z_{ij} &= LF^T = \sum_k l_{ik} f_{kj} \end{aligned} \tag{3.2.19}$$

where $L = (\boldsymbol{l}_1, \cdots, \boldsymbol{l}_K)$ is $N \times K$ loading matrix and $F = (\boldsymbol{f}_1, \cdots, \boldsymbol{f}_K)$ is $P \times K$ factor matrix, and we assume that the prior of $\boldsymbol{l}_1 \cdots, \boldsymbol{l}_K$ and $\boldsymbol{f}_1, \cdots, \boldsymbol{f}_K$ as follows:

$$l_{ik} \sim g_{\boldsymbol{l}_k}(\cdot) \tag{3.2.20}$$

$$f_{kj} \sim g_{\boldsymbol{f}_k}(\cdot) \tag{3.2.21}$$

$g_{\boldsymbol{l}_k}$ and $g_{\boldsymbol{f}_k}$ are from some distribution families, such as $\mathcal{U}$, $\mathcal{S}U$ and $\mathcal{S}N$.

Variational Inference for rank K model

We apply variational EM algorithm to approximate the posterior distribution of $\boldsymbol{l}_k$ and $\boldsymbol{f}_k$ for all $k = 1, \cdots, K$ with the assumption that the approximation of posterior is fully factorized:

$$q(\boldsymbol{l}_1 \cdots, \boldsymbol{l}_K, \boldsymbol{f}_1, \cdots, \boldsymbol{f}_K) = \prod_k q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_{\boldsymbol{f}_k}(\boldsymbol{f}_k).$$

Similarly, our goal is to maximize the ELBO of log likelihood function of multiple factor model. It is hard to maximize the ELBO directly, so we optimize over one tight lower bound of ELBO as follows:

$$
\begin{aligned}
& F(q_{\boldsymbol{l}_1}, \cdots, q_{\boldsymbol{l}_K}, q_{\boldsymbol{f}_1}, \cdots, q_{\boldsymbol{f}_K}, g_{\boldsymbol{l}_1}, \cdots, g_{\boldsymbol{l}_K}, g_{\boldsymbol{f}_1}, \cdots, g_{\boldsymbol{f}_K}) \\
& \geq \int \prod_k q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_{\boldsymbol{f}_k}(\boldsymbol{f}_k) \log \frac{\prod_k p(\boldsymbol{l}_k|g_{\boldsymbol{l}_k}) p(\boldsymbol{f}_k|g_{\boldsymbol{f}_k}) H(L, F, Y, \xi)}{\prod_k q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_{\boldsymbol{f}_k}(\boldsymbol{f}_k)} \\
& \equiv Q(q_{\boldsymbol{l}_1}, \cdots, q_{\boldsymbol{l}_K}, q_{\boldsymbol{f}_1}, \cdots, q_{\boldsymbol{f}_K}, g_{\boldsymbol{l}_1}, \cdots, g_{\boldsymbol{l}_K}, g_{\boldsymbol{f}_1}, \cdots, g_{\boldsymbol{f}_K}, \xi)
\end{aligned}
\tag{3.2.22}
$$

where

$$
H(L, F, Y, \xi) = \exp\left\{ \sum_{ij} \left[\frac{Y_{ij}(\sum_k l_{ik} f_{kj}) - \xi_{ij}}{2} + \log(h(\xi_{ij})) - \tau(\xi_{ij}) \left((\sum_k l_{ik} f_{kj})^2 - \xi_{ij}^2 \right) \right] \right\}. \tag{3.2.23}
$$

To simplify the notations in this section, we denote the evidence lower bound as $F(q, g) \equiv F(q_{\boldsymbol{l}_1}, \cdots, q_{\boldsymbol{l}_K}, q_{\boldsymbol{f}_1}, \cdots, q_{\boldsymbol{f}_K}, g_{\boldsymbol{l}_1}, \cdots, g_{\boldsymbol{l}_K}, g_{\boldsymbol{f}_1}, \cdots, g_{\boldsymbol{f}_K})$, and the tight lower bound as $Q(q, g, \xi) \equiv Q(q_{\boldsymbol{l}_1}, \cdots, q_{\boldsymbol{l}_K}, q_{\boldsymbol{f}_1}, \cdots, q_{\boldsymbol{f}_K}, g_{\boldsymbol{l}_1}, \cdots, g_{\boldsymbol{l}_K}, g_{\boldsymbol{f}_1}, \cdots, g_{\boldsymbol{f}_K}, \xi)$. We apply coordinate ascent to maximize the objective function $Q(q, g, \xi)$, iteratively optimizing over the variational approximate q, and hyper parameter g, for each k holding the others fixed. By plugging the formula of $H(L, F, Y, \xi)$ into the lower bound, $Q(q, g, \xi)$, we can obtain the explicit form:

$$
\begin{aligned}
Q(q, g, \xi) \;=\; & E_q \sum_{ij} \left\{ \frac{Y_{ij}(\sum_k l_{ik} f_{kj}) - \xi_{ij}}{2} + \log(h(\xi_{ij})) - \tau(\xi_{ij}) \left((\sum_k l_{ik} f_{kj})^2 - \xi_{ij}^2 \right) \right\} \\
& + \sum_k E_q \log p(\boldsymbol{l}_k|g_{\boldsymbol{l}_k}) p(\boldsymbol{f}_k|g_{\boldsymbol{f}_k}) - \sum_k E_q \log q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_{\boldsymbol{f}_k}(\boldsymbol{f}_k).
\end{aligned}
$$

From (3.2.8), to maximize the objective function given $q_{\boldsymbol{l}_k}, q_{\boldsymbol{f}_k}, g_{\boldsymbol{l}_k}, g_{\boldsymbol{f}_k}$ for $k = 1, \cdots, K$, we should update ξ as follows:

$$\xi_{ij}^2 = E_q(\sum_k l_{ik} f_{kj})^2 = E_q Z_{ij}^2. \tag{3.2.24}$$

Greedy algorithm

Following the same idea in Appendix A.4, we apply greedy algorithm considering the factor model as an additive model. Greedy algorithm on multiple factors model is a forward procedure by including one factor each time. Details are included in Algorithm 4.

Algorithm 4 Logistic FLASH multiple factors

1: **procedure** GREEDY ALGORITHM

2: **for** $k = 1, \cdots, K$ **do**

3: $t = 0$

4: initialize $q^{(0)}_{\boldsymbol{l}_k}$, $q^{(0)}_{\boldsymbol{f}_k}$, $g^{(0)}_{\boldsymbol{l}_k}$, $g^{(0)}_{\boldsymbol{f}_k}$ and $\xi^2_{(0)}$

5: $\epsilon = 1$

6: **while** $\epsilon > 10^{-5}$ **do**

7: $t = t + 1$

8: $q^{(t)}_{\boldsymbol{l}_k}, g^{(t)}_{\boldsymbol{l}_k} \leftarrow \underset{q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}}{\arg\max} Q(q_{\boldsymbol{l}_k}, q^{(t-1)}_{\boldsymbol{l}_{1\cdots k-1}}, q^{(t-1)}_{\boldsymbol{f}_k}, q^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, g_{\boldsymbol{l}_k}, g^{(t-1)}_{\boldsymbol{l}_{1\cdots k-1}}, g^{(t-1)}_{\boldsymbol{f}_k}, g^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, \xi^2_{(t-1)})$

9: $q^{(t)}_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k(t)} \leftarrow \underset{q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}}{\arg\max} Q(q^{(t)}_{\boldsymbol{l}_k}, q^{(t-1)}_{\boldsymbol{l}_{1\cdots k-1}}, q_{\boldsymbol{f}_k}, q^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, g^{(t)}_{\boldsymbol{l}_k}, g^{(t-1)}_{\boldsymbol{l}_{1\cdots k-1}}, g_{\boldsymbol{f}_k}, g^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, \xi^2_{(t-1)})$

10: $\xi^2_{(t)} \leftarrow \underset{\xi^2}{\arg\max} Q(q^{(t)}_{\boldsymbol{l}_k}, q^{(t)}_{\boldsymbol{l}_{1\cdots k-1}}, q^{(t)}_{\boldsymbol{f}_k}, q^{(t)}_{\boldsymbol{f}_{1\cdots k-1}}, g^{(t)}_{\boldsymbol{l}_k}, g^{(t)}_{\boldsymbol{l}_{1\cdots k-1}}, g^{(t)}_{\boldsymbol{f}_k}, g^{(t)}_{\boldsymbol{f}_{1\cdots k-1}}, \xi^2)$

11: value for stopping criteria:

$$\begin{aligned} \epsilon &= |Q(q^{(t)}_{\boldsymbol{l}_{1,\cdots,k}}, q^{(t)}_{\boldsymbol{f}_{1,\cdots,k}}, \pi^{\boldsymbol{l}_{1,\cdots,k}(t)}, \pi^{\boldsymbol{f}_{1,\cdots,k}(t)}, \xi^2_{(t)}) \\ &- Q(q^{(t-1)}_{\boldsymbol{l}_{1,\cdots,k}}, q^{(t-1)}_{\boldsymbol{f}_{1,\cdots,k}}, \pi^{\boldsymbol{l}_{1,\cdots,k}(t-1)}, \pi^{\boldsymbol{f}_{1,\cdots,k}(t-1)}, \xi^2_{(t-1)})| \end{aligned}$$

12: **if** $g_{\boldsymbol{l}_k} == \delta(0)$ or $g_{\boldsymbol{f}_k} == \delta(0)$ **then**

13: $\tilde{K} = k - 1$

14: **break**

return $list(q_{\boldsymbol{l}_1}, q_{\boldsymbol{f}_1}, \cdots, q_{\boldsymbol{l}_{\tilde{K}}}, q_{\boldsymbol{f}_{\tilde{K}}}; g_{\boldsymbol{l}_1}, g_{\boldsymbol{f}_1}, \cdots, g_{\boldsymbol{l}_{\tilde{K}}}, g_{\boldsymbol{f}_{\tilde{K}}}; \xi^2)$

The step 10 in Algorithm 4 can be solved by (3.2.24). With similar reasoning in section 2.3.3 and Appendix A.3, the optimization over $q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}$ in step 8 of Algorithm

4 can be solved by EB approach of normal means problem with parameters as follows:

$$x_i = \frac{\sum_j [\frac{1}{2} Y_{ij} - 2(\tau(\xi_{ij}) \sum_{t \neq k} (El_{it} Ef_{tj}))] Ef_{kj}}{2 \sum_j (\tau(\xi_{ij}) Ef_{kj}^2)} \tag{3.2.25}$$

$$s_i^2 = \frac{1}{2 \sum_j (\tau(\xi_{ij}) Ef_{kj}^2)}. \tag{3.2.26}$$

Similarly, step 9 of Algorithm 4 can be solved by EB approach of normal means problem with parameters:

$$x_j = \frac{\sum_i [\frac{1}{2} Y_{ij} - 2(\tau(\xi_{ij}) \sum_{t \neq k} (El_{it} Ef_{tj}))] El_{ik}}{2 \sum_j (\tau(\xi_{ij}) El_{ik}^2)} \tag{3.2.27}$$

$$s_j^2 = \frac{1}{2 \sum_i (\tau(\xi_{ij}) El_{ik}^2)}. \tag{3.2.28}$$

To implement this procedure we must initialize the loading and factor for each newly added factor adaptively in Step 4. The initialization of ξ is straightforward, see below. In this procedure, we can't directly get the residual matrix by subtracting the included factors from the observed data matrix as we did in chapter 2, because there is a link function between the low rank structure and the data matrix. Comparing the update between rank one case and rank K case, we choose to use the following residual matrix to get the initialization of factor and loading.

$$\begin{aligned} residual_k &= Y_{ij} - 4[\tau(\xi_{ij}) \sum_{t \neq k} (El_{it} Ef_{tj})] \\ \xi_{ij}^2 &= E_{q_l, q_f} (\sum_{t \neq k} l_{it} f_{tj})^2. \end{aligned}$$

In this repeated procedure, we stop when $g_{\boldsymbol{l}_k}$ and $g_{\boldsymbol{f}_k}$ got optimized at point mass

function, which provides us a rough estimationfor the rank of the underlying structure.

3.3 Binomial Model

In this section, we extend the model for binary data (section 3.2) to a model for binomial data. Bernoulli distribution is a special case of binomial distribution.

3.3.1 Model

We first start with a simple case, rank one model:

$$Y_{ij} = Bin(n_{ij}, p_{ij}) \tag{3.3.1}$$

$$\log \frac{p_{ij}}{1-p_{ij}} = Z_{ij} \tag{3.3.2}$$

$$Z_{ij} = l_i f_j \tag{3.3.3}$$

$$l_i \sim g_{\boldsymbol{l}}(\cdot) \tag{3.3.4}$$

$$f_j \sim g_{\boldsymbol{f}}(\cdot) \tag{3.3.5}$$

where Y_{ij} are the observed data points, $\boldsymbol{l} = (l_i, \cdots, l_N)^T$ is the loading corresponding to the latent factor $\boldsymbol{f} = (f_1, \cdots, f_P)^T$, $g_{\boldsymbol{l}}$ and $g_{\boldsymbol{f}}$ are from some distribution family, such as $\mathcal{U}$, $\mathcal{S}U$ and $\mathcal{S}N$, n_{ij} is known for each $i = 1, \cdots, N, j = 1, \cdots, P$.

3.3.2 Inference

In this section, instead of optimizing the tight lower bound in variational inference as in binary model, we augment this model with a Polya-Gamma random variable (Klami, 2014; Polson et al., 2013) to obtain:

$$P(Y_{ij}, \omega_{ij}|Z_{ij}) = C_{Y_{ij}}^{n_{ij}} 2^{-n_{ij}} e^{\tilde{Y}_{ij} Z_{ij}} e^{-\omega_{ij} Z_{ij}^2/2} PG(\omega_{ij}|n_{ij}, 0) \tag{3.3.6}$$

where $\tilde{Y}_{ij} = Y_{ij} - \frac{n_{ij}}{2}$, $C_{Y_{ij}}^{n_{ij}} = \frac{n_{ij}!}{(n_{ij}-Y_{ij})!Y_{ij}!}$ and $PG(\omega_{ij}|n_{ij}, 0)$ is Polya-Gamma distribution.

Polya-Gamma augmentation

The likelihood function of binomial distribution is:

$$\begin{aligned} p(Y_{ij}|p_{ij}) &= C_{Y_{ij}}^{n_{ij}} p_{ij}^{Y_{ij}} (1-p_{ij})^{(n_{ij}-Y_{ij})} \\ &= C_{Y_{ij}}^{n_{ij}} \frac{(e^{\psi_{ij}})^{Y_{ij}}}{(1+e^{\psi_{ij}})^{n_{ij}}} \end{aligned} \tag{3.3.7}$$

where $\psi_{ij} = logit(p_{ij}) = \log(\frac{p_{ij}}{1-p_{ij}})$. We can use the following results in Polson et al. (2013):

$$\frac{(e^{\psi})^a}{(1+e^{\psi})^b} = 2^{-b} e^{(a-\frac{b}{2})\psi} \int_0^{\infty} e^{-\omega\psi^2/2} p(\omega) d\omega \tag{3.3.8}$$

where $p(\omega) = PG(\omega|b, 0)$ is Polya-Gamma distribution:

$$PG(\omega|b, c) = \cosh^b(c/2)\frac{2^{b-1}}{\Gamma(b)}\sum_{n=0}^{\infty}(-1)^n\frac{\Gamma(n+b)\Gamma(2n+b)}{\Gamma(n+1)\sqrt{2\pi\omega^3}}e^{-\frac{(2n+b)^2}{8\omega}}e^{-\frac{c^2}{2}\omega} \quad (3.3.9)$$

$\cosh(x) = \frac{e^x+e^{-x}}{2}$. This density function is complicated, but we can get the explicit form of expectation $E\omega = \frac{b}{2c}\tanh(c/2)$ and $\tanh(x) = \frac{e^x-e^{-x}}{e^x+e^{-x}}$.

Variational Inference

We apply variational inference (Bishop., 2006) with mean-field assumption. The purpose of this variational approach is to maximize the lower bound of log likelihood function as follows:

$$\begin{aligned}
\log P(Y) &= \log \iiint p(Y, \omega|\boldsymbol{l}, \boldsymbol{f})p(\boldsymbol{l}|g_{\boldsymbol{l}})p(\boldsymbol{f}|g_{\boldsymbol{f}})d\boldsymbol{f}d\boldsymbol{l}d\omega \\
&\geq \iiint q_{\boldsymbol{f}}(\boldsymbol{f})q_{\boldsymbol{l}}(\boldsymbol{l})q_\omega(\omega)\log\frac{p(Y, \omega|\boldsymbol{l}, \boldsymbol{f})p(\boldsymbol{l}|g_{\boldsymbol{l}})p(\boldsymbol{f}|g_{\boldsymbol{f}})}{q_{\boldsymbol{l}}(\boldsymbol{l})q_{\boldsymbol{f}}(\boldsymbol{f})q_\omega(\omega)}d\boldsymbol{f}d\boldsymbol{l}d\omega \\
&\equiv F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, q_\omega) \quad (3.3.10)
\end{aligned}$$

where ω is $N \times P$ matrix. We first derive the variational updates for elements of ω:

$$\begin{aligned}
q_{\omega_{ij}}(\omega_{ij}) &\propto \exp\left\{E_{q_{\boldsymbol{l}},q_{\boldsymbol{f}}}log(p(Y_{ij}, \omega_{ij}|\boldsymbol{l}, \boldsymbol{f})p(\boldsymbol{l}|g_{\boldsymbol{l}})p(\boldsymbol{f}|g_{\boldsymbol{f}}))\right\} \quad (3.3.11) \\
&= \exp\left\{-\frac{E_q l_i^2 E_q f_j^2\omega_{ij}}{2} + \log PG(\omega_{ij}|n_{ij}, 0)\right\} \\
&\propto PG(\omega_{ij}|n_{ij}, \sqrt{\xi_{ij}})
\end{aligned}$$

where $\xi_{ij} = \sqrt{E_q l_i^2 E_q f_j^2}$ and $q(\omega) = \prod_{ij} q_{\omega_{ij}}(\omega_{ij})$. Under this variational approximation for ω, we can get the first moment:

$$E_q \omega_{ij} = \frac{n_{ij}}{2\xi_{ij}} \tanh(\frac{\xi_{ij}}{2}). \tag{3.3.12}$$

To get the variational approximation of posterior distribution of l_i for each $i = 1, \cdots, N$, we maximize the lower bound of log likelihood over $q_{\boldsymbol{l}}, g_{\boldsymbol{l}}$:

$$\begin{aligned} F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, q_\omega) &= E_q \log(p(Y, \omega|\boldsymbol{l}, \boldsymbol{f}) p(\boldsymbol{l}|g_{\boldsymbol{l}}) p(\boldsymbol{f}|g_{\boldsymbol{f}})) \qquad (3.3.13) \\ &\quad - E_q q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f}) + C_l \\ &= \sum_{ij} \left\{ -\frac{l_i^2 E_q f_j^2 E_q \omega_{ij}}{2} + \tilde{Y}_{ij} l_i E_q f_j \right\} \\ &\quad - E_q q_{\boldsymbol{l}}(\boldsymbol{l}) + E_q p(\boldsymbol{l}|g_{\boldsymbol{l}}) + C_l' \end{aligned} \tag{3.3.14}$$

where C_l and C_l' are constants with respect to $\boldsymbol{l}$, and

$$E_q \omega_{ij} = \frac{n_{ij}}{2\xi_{ij}} \tanh(\frac{\xi_{ij}}{2}).$$

Based on Lemma 1 in Appendix A.1, we can apply EB approach for normal means

problem to solve this optimization problem with parameters:

$$x_i = \frac{\sum_j (\tilde{Y}_{ij} E f_j)}{\sum_j (E_q \omega_{ij} E f_j^2)} \tag{3.3.15}$$

$$s_i^2 = \frac{1}{\sum_j (E_q \omega_{ij} E f_j^2)} \tag{3.3.16}$$

We notice that the updates in (3.3.15)-(3.3.16) are the same as the ones in (3.2.25)-(3.2.26) when fixing $n_{ij} = 1$ and transforming Y_{ij} into the same scale in both procedures. It is interesting that we undertake different procedures, lower bound method and Polya-Gamma augmentation, which provides the same updates in the variational algorithms.

Similarly, the updates for $f_j, j = 1, \cdots, P$ can be obtained from EB approach for normal means problem with parameters:

$$x_j = \frac{\sum_i (\tilde{Y}_{ij} E l_i)}{\sum_i (E_q \omega_{ij} E l_i^2)} \tag{3.3.17}$$

$$s_j^2 = \frac{1}{\sum_i (E_q \omega_{ij} E l_i^2)} \tag{3.3.18}$$

Here we find that the updates for $\boldsymbol{l}, \boldsymbol{f}, \xi$ using Polya-Gamma augmentation in binomial model, with $n_{ij} = 1$ fixed, (3.3.15)-(3.3.18), are the same as the updates in binary model applying lower bound method in variational inference, (3.2.14)-(3.2.18). The inference strategies in two different models are both variational inference as well, so the the implementations of algorithms are exactly the same. We can just use Algorithm 3 to implement our approach.

3.3.3 Rank K model

In this section, we consider a more general model with multiple factors:

$$Y_{ij} = Bin(n_{ij}, p_{ij}) \tag{3.3.19}$$

$$\log \frac{p_{ij}}{1-p_{ij}} = Z_{ij} \tag{3.3.20}$$

$$Z_{ij} = \sum_k l_{ik} f_{kj} \tag{3.3.21}$$

where we assume the same prior as the binary model, $L = (\boldsymbol{l}_1, \cdots, \boldsymbol{l}_K)$ is $N \times K$ the loading matrix and $F = (\boldsymbol{f}_1, \cdots, \boldsymbol{f}_K)$ is $P \times K$ the factor matrix, and the prior of $\boldsymbol{l}_1 \cdots, \boldsymbol{l}_K$ and $\boldsymbol{f}_1, \cdots, \boldsymbol{f}_K$ as follows:

$$l_{ik} \sim g^k_{\boldsymbol{l}}(\cdot) \tag{3.3.22}$$

$$f_{kj} \sim g^k_{\boldsymbol{f}}(\cdot) \tag{3.3.23}$$

$g^k_{\boldsymbol{l}}$ and $g^k_{\boldsymbol{f}}$ are from some distribution families, such as $\mathcal{U}$, $\mathcal{S}U$ and $\mathcal{S}N$. We apply Polya-Gamma augmentation to the multiple factor model and use variational inference to optimize the objective function:

$$\begin{aligned} &F(q_{\boldsymbol{l}_{1,\cdots,K}}, g_{\boldsymbol{l}_{1,\cdots,K}}, q_{\boldsymbol{f}_{1,\cdots,K}}, g_{\boldsymbol{f}_{1,\cdots,K}}, q_\omega) \qquad (3.3.24)\\ &\equiv \iiint \prod_k q_{\boldsymbol{f}_k}(\boldsymbol{f}) q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_\omega(\omega) \log \frac{p(Y, \omega | L, F) \prod_k \prod_k p(\boldsymbol{l}_k | g_{\boldsymbol{l}_k}) p(\boldsymbol{f}_k | g_{f_k})}{\prod_k q_{\boldsymbol{f}_k}(\boldsymbol{f}) q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_\omega(\omega)} d\boldsymbol{f} d\boldsymbol{l} d\omega \end{aligned}$$

More details of the variational algorithm are included in Appendix B.1.

3.4 Numerical Studies

3.4.1 Simulations for binary data sets

We compare Logistic FLASH and FLASH on the simulated data sets in performance of latent structure estimation. In this section we focus on rank one model, where we simulate data sets with $N = 200$ and $P = 300$ following the model:

$$\begin{aligned} Y_{ij} &= 2 * Bern(p_{ij}) - 1 \\ \log \frac{p_{ij}}{1 - p_{ij}} &= Z_{ij} \\ Z_{ij} &= l_i f_j \\ l_i, f_j &\sim \pi_0 \delta_0 + (1 - \pi_0) \left(\sum_{k=1}^{4} \frac{1}{4} N(0, \sigma_k^2) \right). \end{aligned}$$

In this part, we simulate three different situations:

1. Dense factors and loadings with $(\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (0.1, 0.5, 1, 3)$ and $(\pi_0, \pi_1, \pi_2, \pi_3, \pi_4) = (0, 0.3, 0.2, 0.1, 0.4)$, in Figure 3.1 (row 1).

2. Intermediate sparse factors and loadings with $(\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (0.1, 0.5, 1, 3)$ and $(\pi_0, \pi_1, \pi_2, \pi_3, \pi_4) = (0.5, 0.025, 0.025, 0.025, 0.425)$, in Figure 3.1 (row 2).

3. Sparse factors and loadings with $(\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (0.1, 0.5, 1, 10)$ and $(\pi_0, \pi_1, \pi_2, \pi_3, \pi_4) = (0.7, 0.015, 0.015, 0.015, 0.255)$, in Figure 3.1 (row 3).

To make the results comparable, we transform the result of Logistic FLASH to the same scale as the result from FLASH. We apply FLASH to data matrix Y, so the estimation should be the low rank estimation of EY, which is a matrix of $2p_{ij} - 1$

with $p_{ij} = P(Y_{ij} = 1)$. We apply the following transformation to get estimation for $2p_{ij} - 1$:

$$2p_{ij} - 1 = 2logit^{-1}(\hat{l}_i \hat{f}_j) - 1 \tag{3.4.1}$$

where $\hat{l}_i, \hat{f}_j$ are the estimation from Logistic FLASH. In Figure 3.1, we plot the estimation of $2p_{ij} - 1$ against its real value and we can see that Logistic FLASH has better performance in all these three situations as the estimation from Logistic FLASH are closer to the identical line. To quantify the performance, we use the following RMSE as criteria comparison:

$$RMSE = \frac{\sqrt{\frac{1}{NP}\sum_{ij}[(2p_{ij} - 1) - (2\hat{p}_{ij} - 1)]^2}}{\sqrt{\frac{1}{NP}\sum_{ij}[(2p_{ij} - 1) - 0]^2}} \tag{3.4.2}$$

In Figure 3.1, the RMSE of FLASH and Logistic FLASH in case 1 are: 0.343 and 0.312; in case 2 are: 0.329 and 0.294; in case 3 are: 0.361 and 0.309.

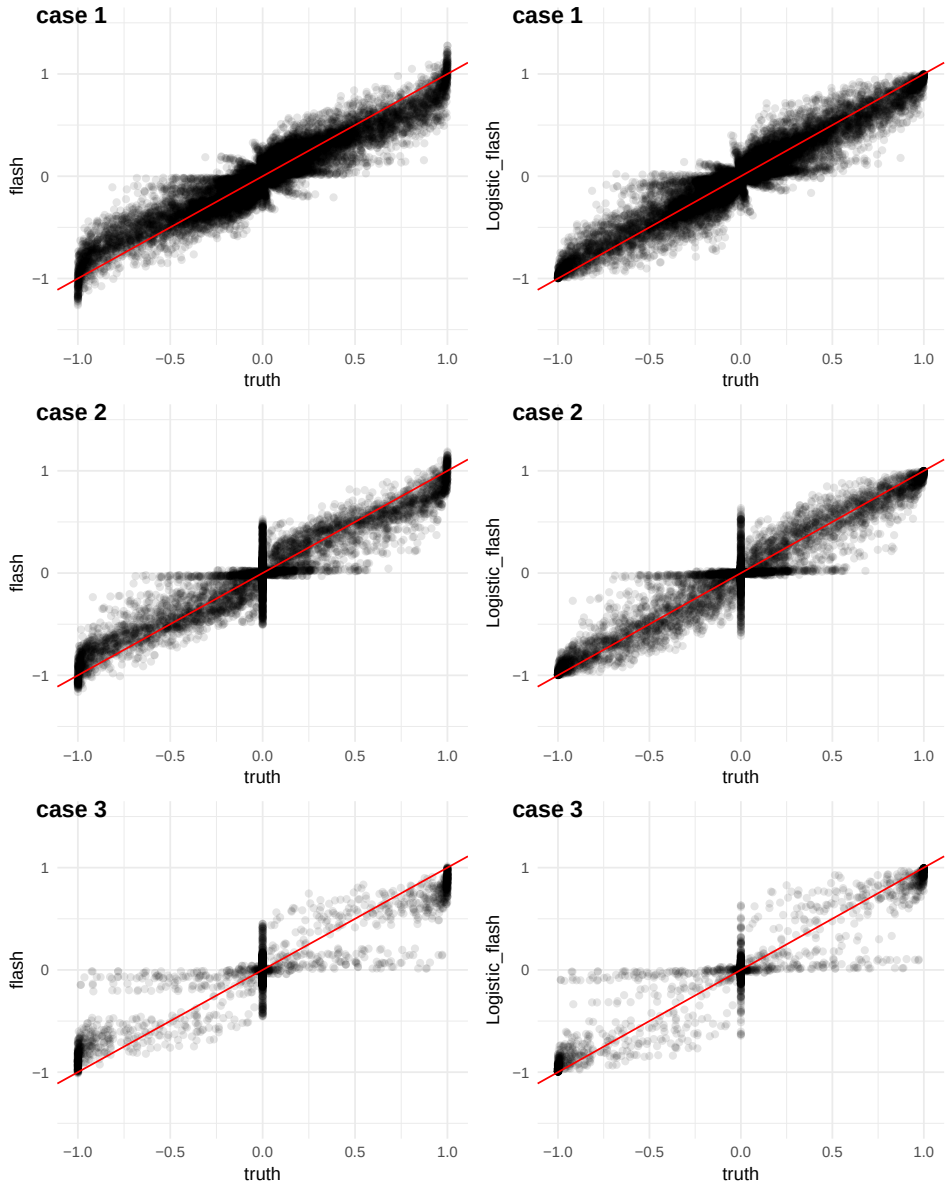

Figure 3.1: This figure shows the comparison between FLASH and Logistic FLASH in binary data sets. The left one in each row shows the estimation of $(2p_{ij} - 1)$ by FLASH, and the right part is the estimation $(2p_{ij} - 1)$ by Logistic FLASH.

3.4.2 Simulations for Binomial data sets

In this section, we compare Logistic FLASH and FLASH on simulated Binomial data sets with rank one model. We simulate data sets with $N = 200$ and $P = 300$ following model:

$$\begin{aligned} n_{ij} &= Poisson(\lambda) + 1 \\ Y_{ij} &= Bin(n_{ij}, p_{ij}) \\ \log \frac{p_{ij}}{1 - p_{ij}} &= Z_{ij} \\ Z_{ij} &= l_i f_j \\ l_i, f_j &\sim \pi_0 \delta_0 + (1 - \pi_0) \left(\sum_{k=1}^{4} \frac{1}{4} N(0, \sigma_k^2) \right) \end{aligned}$$

where $\lambda = 5$. We add one to the Poisson random variable in order to make sure n_{ij} to be positive and we subtract $\frac{n_{ij}}{2}$ from the Binomial random variable to make it easy to apply FLASH. In this part, we simulate three different situations as well:

1. Dense factors and loadings with $(\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (0.1, 0.5, 1, 3)$ and $(\pi_0, \pi_1, \pi_2, \pi_3, \pi_4) = (0, 0.3, 0.2, 0.1, 0.4)$, in Figure 3.2 (row 1).

2. Intermediate sparse factors and loadings with $(\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (0.1, 0.5, 1, 3)$ and $(\pi_0, \pi_1, \pi_2, \pi_3, \pi_4) = (0.5, 0.025, 0.025, 0.025, 0.425)$, in Figure 3.2 (row 2).

3. Sparse factors and loadings with $(\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (0.1, 0.5, 1, 10)$ and $(\pi_0, \pi_1, \pi_2, \pi_3, \pi_4) = (0.8, 0.01, 0.01, 0.01, 0.17)$, in Figure 3.2 (row 3).

We transform both results into $2p_{ij} - 1$ with $p_{ij} = P(Y_{ij} = 1)$ corresponding to

the previous example. In FLASH, we apply factor analysis to data matrix $\tilde{Y}$, where $\tilde{Y}_{ij} = \frac{Y_{ij} - \frac{n_{ij}}{2}}{\frac{n_{ij}}{2}}$. The estimation should be the low rank estimation of $E\tilde{Y} = (2\boldsymbol{P} - 1)$, where each element of $\boldsymbol{P}$ is $p_{ij} = P(Y_{ij} = 1)$. We apply the same approach as that in the previous simulation study of binary data sets to get estimation of $2\boldsymbol{P} - 1$ in Logistic FLASH. We plot estimated $2\boldsymbol{P} - 1$ from different methods against the true value in Figure 3.2. From the Figure, we can see that Logistic FLASH, which estimation is closer to the identical line, has better performance in these simulation studies. To quantify the performance, we use the RMSE in (3.4.2). In Figure 3.2, the RMSE of FLASH and Logistic FLASH in case 1 are: 0.203 and 0.109; in case 2 are: 0.192 and 0.130; in case 3 are: 0.192 and 0.155.

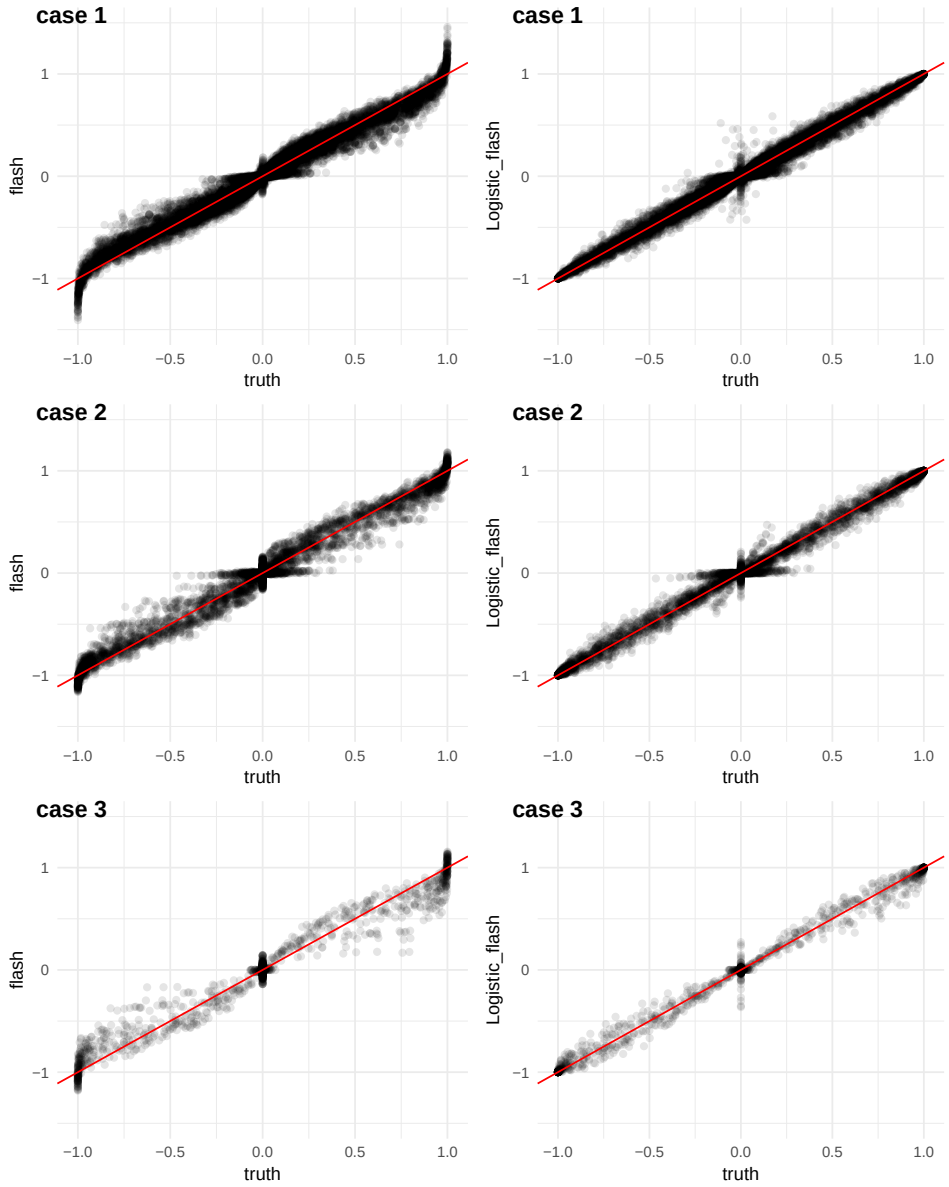

Figure 3.2: This figure shows the comparison between FLASH and Logistic FLASH in Binomial data sets. The left one in each row shows the estimation of $(2p_{ij} - 1)$ by FLASH, and the right part is the estimation $(2p_{ij} - 1)$ by Logistic FLASH.

3.4.3 Real data

We apply Logistic FLASH to allele frequency data with 8,000 columns and 26 rows to understand the underlying structure of different populations. The original data is from 1000 genomes project (Consortium et al., 2015a). We assume that the data follows binomial distribution, so we apply binomial version of Logistic FLASH. In the result of greedy algorithm, there are 14 factors selected. We assign the populations in to five different groups and the group information is from http://www.internationalgenome.org/category/population/ . Details of the populations and group information are in Table 3.1.

In Figure 3.3, we plot the first 6 factor loadings using the same color for the populations in each group. All the populations loaded on the first factor, so this factor can be interpreted as a common factor. Factors 2-6 have clear interpretations according to the loading values. Factor 2 can be interpreted as specific factor for African populations; Similarly, Factor 3, 4, 5 can be interpreted as corresponding factors for East Asian, European and South Asian populations respectively; Factor 6 can be interpreted as specific factor for MXL (Mexican Ancestry from Los Angeles USA) and PEL (Peruvians from Lima, Peru). We can see that the populations in group 4 are loaded on other factors. For example, MXL and PEL load on "European" factor and "Asian" factors, and CLM (Colombians from Medellin, Colombia) and PUR (Puerto Ricans from Puerto Rico) have very big loading values on "European" factor. The interpretable factors provide clustering information of populations. To get the clustering structure from our results, we make a heatmap of the estimated low rank structure excluding the common factor (factor 1) in Figure 3.4. We can see

Table 3.1: Description of populations

Population	*Description*	*Group*
CHB	Han Chinese in Bejing, China	1
JPT	Japanese in Tokyo, Japan	1
CHS	Southern Han Chinese	1
CDX	Chinese Dai in Xishuangbanna, China	1
KHV	Kinh in Ho Chi Minh City, Vietnam	1
CEU	Utah Residents (CEPH) with Northern and Western European Ancestry	2
TSI	Toscani in Italia	2
FIN	Finnish in Finland	2
GBR	British in England and Scotland	2
IBS	Iberian Population in Spain	2
YRI	Yoruba in Ibadan, Nigeria	3
LWK	Luhya in Webuye, Kenya	3
GWD	Gambian in Western Divisions in the Gambia	3
MSL	Mende in Sierra Leone	3
ESN	Esan in Nigeria	3
ASW	Americans of African Ancestry in SW USA	3
ACB	African Caribbeans in Barbados	3
MXL	Mexican Ancestry from Los Angeles USA	4
PUR	Puerto Ricans from Puerto Rico	4
CLM	Colombians from Medellin, Colombia	4
PEL	Peruvians from Lima, Peru	4
GIH	Gujarati Indian from Houston, Texas	5
PJL	Punjabi from Lahore, Pakistan	5
BEB	Bengali from Bangladesh	5
STU	Sri Lankan Tamil from the UK	5
ITU	Indian Telugu from the UK	5

that the populations are almost clustered in the same way as the group information in Table 3.1.

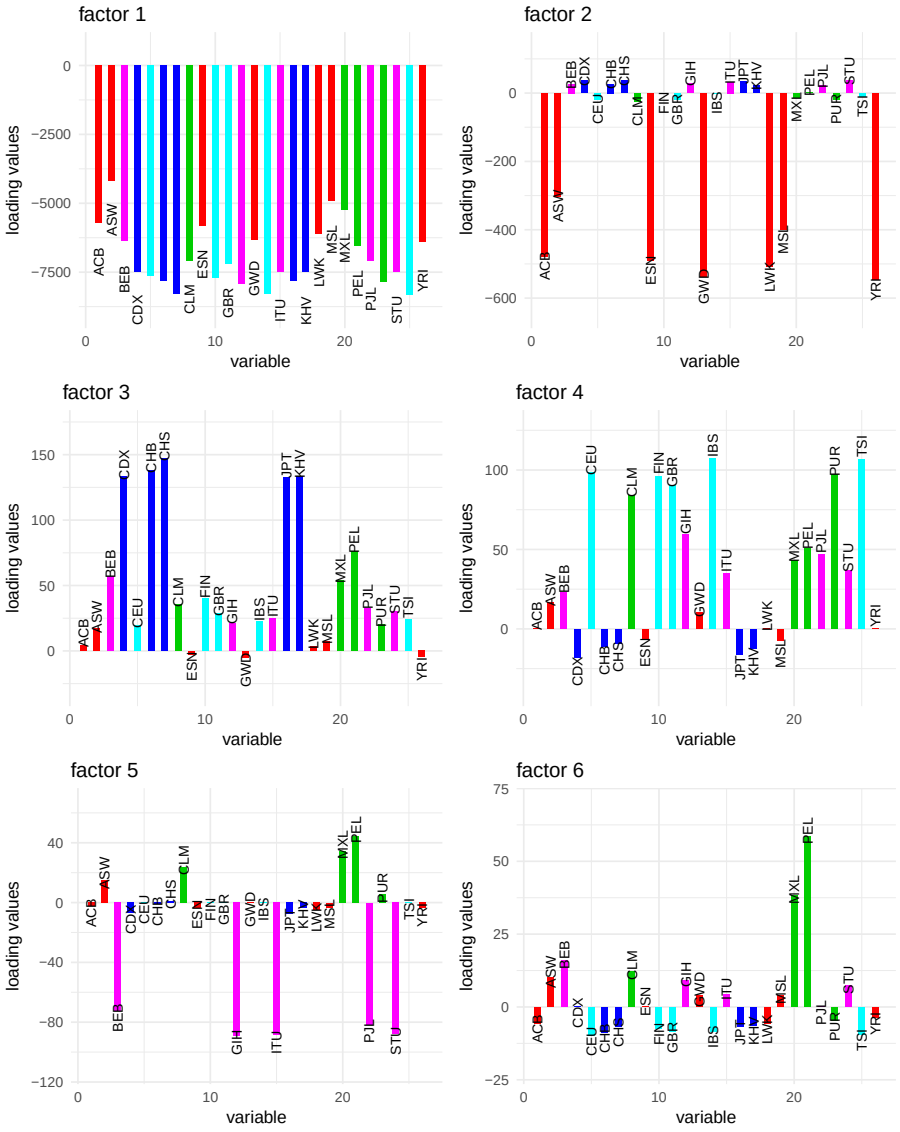

Figure 3.3: Factor loadings for the first 6 factors. Group 1 in blue; Group 2 in light blue; Group 3 in red; Group 4 in green; Group 5 in magenta, where group information is in Table 3.1.

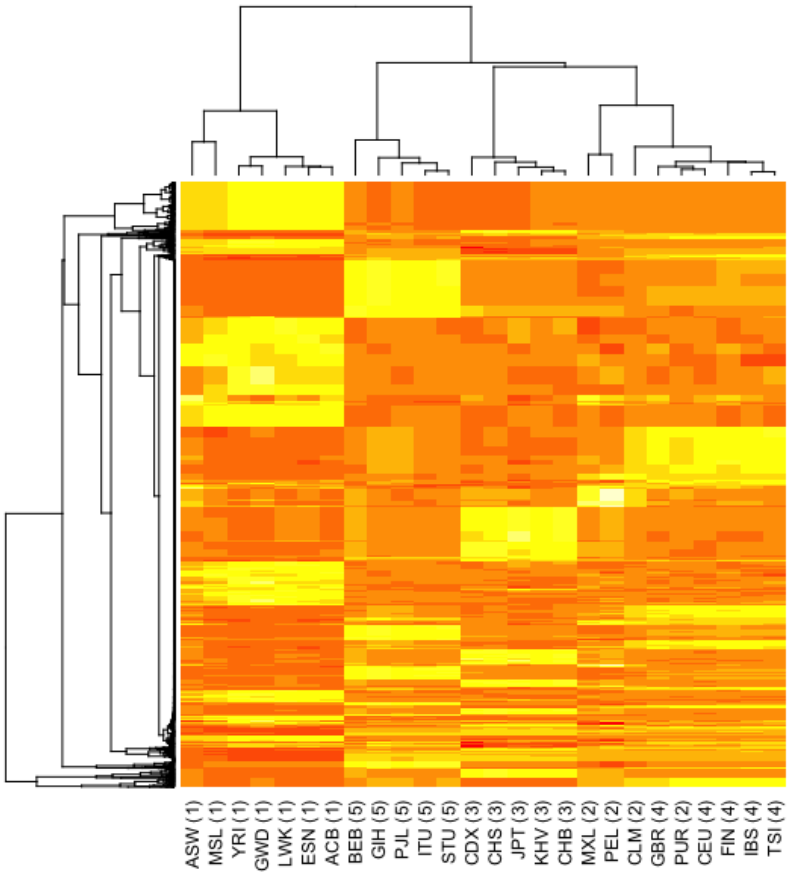

Figure 3.4: Heatmap of the low rank estimation excluding factor 1.

3.5 Discussion

One interesting extensions is to find a model for more general count data. In this situation, we can try to use Poisson distribution to model the counts as follows:

$$Y_{ij}|\lambda_{ij}, p_{ij} \sim Poisson(\lambda_{ij}p_{ij}) \tag{3.5.1}$$

$$\log \frac{p_{ij}}{1-p_{ij}} = l_i f_j \tag{3.5.2}$$

where we assume that the priors for $\boldsymbol{l}, \boldsymbol{f}, \lambda$ are:

$$\begin{aligned} \lambda_{ij} &\sim g_\lambda(\cdot) \\ l_i &\sim g_{\boldsymbol{f}}(\cdot) \\ f_j &\sim g_{\boldsymbol{l}}(\cdot). \end{aligned}$$

$g_{\boldsymbol{l}}$ and $g_{\boldsymbol{f}}$ are from some distribution families, such as $\mathcal{U}$, $\mathcal{S}U$ and $\mathcal{S}N$, and g_λ is from some other distribution families in which the random variable takes positive values. More details are included in Appendix B.2

CHAPTER 4

A COMPARISON OF STATISTICAL METHODS FOR ESTIMATING PRECISION MATRICES

4.1 Introduction

Estimation of covariance matrix and precision matrix are fundamental problems in modern statistics and widely used in many fields, such as classification, discriminant analysis and graphical model, with applications in many areas. In many multivariate statistical analyses, we need the estimation of precision matrix to calculate the likelihood based on multivariate normal assumption. We denote $\mathbf{X}_{N\times P} = (X_1, \cdots, X_N)^T$ as data matrix. Drton and Perlman (2007) proposed multiple testing procedure to estimate sparse precision matrix with $N > P$. In practice, estimating the precision matrix is challenging for large-scale data sets, especially when the number of variables is much larger than the sample size, $P > N$.

In this chapter, we focus on the situations where the dimension of data is larger than the sample size. For covariance matrix, a natural and straightforward estimation is sample covariance matrix, $\frac{1}{N}\mathbf{X}^T\mathbf{X}$, which is the maximum likelihood estimator (MLE) under multivariate normal distribution. Although there are many good properties as of MLE, this estimation is singular when N is smaller than P. To estimate the precision matrix, we can't directly inverse $\hat{\Sigma}$ which is not invertible, and it's also difficult to estimate $\frac{P(P+1)}{2}$ parameters in Σ or Ω using N samples. In the setting where P is much larger than N, the accuracy of the estimation of covariance/precision matrix and the computational efficiency are the main challenges in

practical studies. We need to assume some special structures for covariance matrix or precision matrix to to reduce the number of free parameters.

There are many possible methods based on different assumptions on the structure of precision matrix or covariance matrix; See Fan et al. (2016) for a recent review. We mainly consider two assumptions: 1, sparse assumption on precision matrix Ω; 2, low-rank assumption on the data matrix $\mathbf{X}$. Methods based on sparse assumption on Ω (Friedman and Tibshirani, 2008; Liu and Wang, 2017; Cai et al., 2011) are commonly used. However, in many applications, directly assuming the precision matrix might not be proper, especially when some latent factors are shared across variables. For example, gene expression might be co-regulated by transcript factors as we discussed in chapter 2. Methods based on low rank assumption apply factor analysis to understand the latent structure of data matrix. In practice, it is hard to know the real structure of the precision matrix, so the choice of methods based on different types of assumptions affects the performance of the estimations.

Besides the choice of methods under different assumptions, the performance of precision matrix estimation also depends on the selection of tuning parameters within the chosen methods. For example, the sparsity level of estimated precision matrix depends on the tuning parameter of the penalty term in penalized likelihood methods. Theoretical results for choosing the optimal tuning parameter usually depend on the real value of Ω which is unobserved in practice. There are different criterions to provide data-driven way to choose the optimal value, such as Bayesian information criterion (BIC, Schwarz (1978); Foygel and Drton (2010); Yuan (2010)), cross validation (CV,Efron (1982)), Bi-cross-validation (BCV, Owen and Perry (2009)), rotation

information criterion (RIC, Lysen (2009)) and stability approach for regularization selection (StARS, Liu et al. (2010)).

In this chapter, we apply a empirical benchmarking system to make data-driven suggestions to estimate the precision matrix by choosing proper methods under some specific assumptions. This empirical benchmarking system is currently developed Stephens Lab, which is called Dynamic Statistical Comparisons.

4.2 Dynamic Statistical Comparisons

It is hard to say which assumption is more reasonable for specific data in different situations. Despite theoretical and empirical results in individual papers, it is still challenging to decide in practice which methods to use. For example, in our procedure in estimating the precision matrix Ω, there are numerous methods based on different assumptions as we mentioned above. In real data analysis, the covariance structure of different data sets are complex and unobserved, so that it is difficult to pick the best method theoretically. What's more, for some specific methods we choose, there are some tuning parameters, such as the tuning parameters in penalized likelihood methods, to determine the penalty terms. Some theoretical results of solving optimal tuning parameters depend on the true value of the precision matrix, which we can't observe in practice, so we need empirical procedure to select the optimal tuning parameters based on some criterion, such as BIC and cross-validation(CV). Most of the researches are methodology-oriented, which develop new methods and apply them to real data sets with comparisons between proposed methods and existing approaches in literatures. In practice, we are more interested in finding the optimal method

for data analysis on specific data sets. Therefore, we need an empirical benchmarking procedure to provide data-driven suggestions on selecting optimal methods with proper tuning parameters for specific data we are interested in.

Dynamic Statistical Comparisons (DSC) is a empirical benchmarking system to help researchers perform statistical comparisons among methods and data sets. In developing new statistical methods, it is very common to compare the new methods with existing ones on the same problem, in order to assess the performance of different methods. However, the comparisons are usually inefficient in many ways. Different comparisons requires familiarization with methods, creation of pipelines, details of implementation in the softwares and details to make sure running the scripts in right way, which makes the comparison some times incredibly time-consuming. And the comparisons are easily out of date in some fast-moving fields where new methods comes out frequently. This situation almost inevitably leads to waste a large amount efforts from different research groups. DSC aims to avoid wasting those efforts in inefficient comparisons. A DSC object contains a public internet repository that allows researchers to compare various methods on larger number of data sets in a more general and productive way. DSC makes it easy to add new methods and experiments with new settings of existing methods into current comparison, and also makes it easy to add new data sets into comparison as well. In this chapter, we use DSC procedure to compare different methods to estimate covariance matrix and precision matrix. More details of DSC are provided in

$$https://stephenslab.github.io/dsc-wiki/.$$

4.3 Methods

We denote $\mathbf{X}_{N\times P} = (X_1, \cdots, X_N)^T$ as data matrix with each column following multivariate distribution, $X_i \sim N(0, \Sigma_{P\times P})$, $i = 1, \cdots, N$; $\Omega = \Sigma^{-1}$ as the precision matrix.

4.3.1 Methods based on sparse graph assumption

As precision matrix encodes the graph under Gaussian distribution, the sparsity of the edges in graph is equivalent to assuming the precision matrix to be sparse. We denote $\mathbf{x}$ as a random vector following multivariate Gaussian distribution, $\mathbf{x} \sim N(0, \Sigma_{P\times P})$. In the setting of graphical model (Lauritzen, 1996), undirected graphs $G = (V, E)$ describes conditional independence among variables of interest, where V is the set of vertex in the graph and E is the set of the edges. G encodes the conditional independence among the elements of the random vector $\mathbf{x}$, $\mathbf{x}_j; j = 1, \cdots, P$, where each element $\mathbf{x}_j$ represents a node in vertex set V and $(i, j) \in E$ implies nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected. $(i, j) \notin E$ represents that $\mathbf{x}_i$ and $\mathbf{x}_j$ are conditionally independent given other variables, $\mathbf{x}_{-(i,j)}$. As we know, in multivariate Gaussian distribution, the partial correlation between $\mathbf{x}_i$ and $\mathbf{x}_j$ equal to zero is the sufficient and necessary condition for conditional independence, which is corresponding to $\Omega_{ij} = 0$:

$$(i, j) \in E \iff \Omega_{ij} \neq 0. \tag{4.3.1}$$

The estimation of precision matrix Ω under sparse assumption is of significant

interest. There are many approaches introduced in literatures, see Fan et al. (2016) for a recent review. In this chapter, we consider a selective set of methods from different types (Friedman and Tibshirani, 2008; Liu and Wang, 2017; Cai et al., 2011).

Glasso

Graphical Lasso (Glasso, Friedman and Tibshirani (2008)) is a penalized likelihood method. Penalized likelihood method is one of the most widely used approaches. Friedman and Tibshirani (2008); Yuan and Lin (2007); Banerjee et al. (2008); Fan et al. (2009) proposed methods based on penalized likelihood. The theoretical properties of the penalized likelihood methods are studied in Ravikumar et al. (2011); Rothman et al. (2008); Lam and Fan (2009) when dimension P is larger than sample size N.

The log-likelihood of the data matrix, $\mathbf{X}$ is as follows:

$$\log(L(\Omega; \mathbf{X})) = \log|\Omega| - \mathbf{tr}(S\Omega) + C_0 \tag{4.3.2}$$

where C_0 is a constant with respect to Ω, $S = \frac{1}{N}\mathbf{X}^T\mathbf{X}$ which is sample covariance matrix, $\mathbf{tr}(\cdot)$ is the trace of matrix and $|\Omega|$ is the determinant of Ω. Glasso considers the following optimization problem to maximize the penalized log-likelihood function:

$$\hat{\Omega} = \arg\max_{\Omega}\{\log|\Omega| - \mathbf{tr}(S\Omega) - \rho||\Omega||_1\} \tag{4.3.3}$$

where ρ is tuning parameter to control the sparsity level of $\hat{\Omega}$, and $||\cdot||_1$ is l_1 norm

for matrix. In more general cases, we can use other penalty functions to encourage sparsity on $\hat{\Omega}$ (Shen et al., 2012).

TIGER

Tuning-insensitive precision matrix estimation (TIGER, Liu and Wang (2017)) estimates Ω using column by column regression based on partial correlation. Partial correlation based methods use the relationship conditional distribution of multivariate normal and linear regression to infer the relationship between elements in precision matrix and conditional correlation. More specifically, they consider the regression models to regress each variable ($i = 1 \cdots P$) against to all others:

$$X_i = \sum_{j \neq i} \beta_j^i X_j + \epsilon_i \tag{4.3.4}$$

$$\epsilon_i \sim N(0, \sigma_{\epsilon i}^2) \tag{4.3.5}$$

The relationship between the regression coefficient and the elements in precision matrix is:

$$\beta_j^i = -\Omega_{ij}/\Omega_{ii} \tag{4.3.6}$$

$$\sigma_{\epsilon_i}^2 = 1/\Omega_{ii} \tag{4.3.7}$$

The main idea of this type of procedure is to apply the solution of the regression problem to obtain the estimation the precision matrix for each column. To solve the regression problem, there are many different procedures, such as Lasso, Dantzig

selector and SQRT-lasso to get certain sparsity for the column of precision matrix. Meinshausen and Bhlmann (2006) proposes a neighborhood selection scheme by using Lasso to the regression problem as variable selection approach:

$$\hat{\beta}^i = \arg\min ||X_i - \sum_{j \neq i} \beta_j^i X_j||_2^2 + \rho_i \sum_{j \neq i} |\beta_j^i|$$

and apply the estimations to 4.3.6. Partial Correlation Screening (Huang et al., 2016) is also a partial correlation based method, which uses orders of partial correlation including the variables one by one with a cleaning step following. Yuan (2010) also provided a approach based on the conditional correlation applying Dantzig selector:

$$\begin{aligned} \hat{\beta}^i &= \arg\min ||\beta^i||_1 \\ \text{subject to} \quad & ||S_{-i,i} - S_{-i,-i}\beta^i||_\infty \leq \rho_i \end{aligned} \tag{4.3.8}$$

where S is still the sample covariance matrix and we can apply the estimations to 4.3.6 to get estimation of Ω. The methods mentioned above depend on tuning parameters to control bias variance trade-off. Although they provide theoretical choice of the parameters, it is hard to apply those in real data. It is because theoretical tuning parameters always depend on the true value of Ω.

TIGER is asymptotically tuning-free method, which is also a partial correlation based method. The main difference between this approach and the methods mentioned above is it applies SQRT-lasso (Belloni et al., 2011) to solve the penalized regression problem. SQRT-lasso is a penalized likelihood method for regression

problem with L_1 penalty:

$$\beta = \arg\min\{\frac{1}{n}||y - x\beta||_2 + \lambda||\beta||_1\} \tag{4.3.9}$$

where λ is the tuning parameter. In the objective function, it uses $||y - x\beta||_2$ rather than $||y - x\beta||_2^2$ in Lasso. Unlike Lasso and Dantzig selector, the choice of tuning parameter, λ, doesn't depend on the real value of unknown parameters asymptotically.

To make the final estimation symmetric, we can apply $\min(\hat{\Omega}_{ij}, \hat{\Omega}_{ji})$ or $\frac{\hat{\Omega}+\hat{\Omega}'}{2}$ to the estimation from regression coefficients.

CLIME

A Constrained l_1 Minimization Approach to Sparse Precision Matrix Estimation (CLIME, Cai et al. (2011)) introduce sparsity directly on columns of Ω and treat the sub-problem of each column as linear program. The estimation is the solution of the following optimization problem:

$$\begin{aligned} \hat{\Omega} &= \arg\min : ||\Omega||_1 \\ \text{subject to} &\quad ||S\Omega - I||_\infty \leq \lambda_n \end{aligned} \tag{4.3.10}$$

where λ_n is a tuning parameter. In the above approach there is no constraint to make $\hat{\Omega}$ symmetric, so the final solution is taking the smaller one from each pair: $\hat{\Omega}_{ij}, \hat{\Omega}_{ji}$. And then this convex problem can be decomposed into P vector minimiza-

tion problems.

$$\begin{aligned} \hat{\Omega}_j &= \arg\min : ||\Omega_j||_1 \\ \text{subject to} &\quad ||S\Omega_{.j} - e_j||_\infty \leq \lambda_n \end{aligned} \tag{4.3.11}$$

where e_j is vector with j^{th} element being 1 and others being 0. The estimation of $\hat{\Omega}$ is obtained by combine all the column estimations: $(\hat{\Omega}_1, \cdots, \hat{\Omega}_P)$. The final estimation is obtained after symmetrization method. The authors prove that the symmetric $\hat{\Omega}$ from clime is asymptotically positive definite. Liu and Luo (2015) proposed similar approach, SCIO estimator.

4.3.2 Methods based on low rank assumption

Instead of assuming the precision matrix is sparse, we can assume that the data matrix has low rank structure. By applying factor analysis in the context of estimation covariance matrix and precision matrix, we first map the data matrix into a low-dimension space which maintains most the variation information, and then we use this low rank structure to estimate precision matrix. Factor model can be described as follows:

$$\mathbf{X}_{N\times P} = L_{N\times K} F_{K\times P} + E_{N\times P} \tag{4.3.12}$$

where $\mathbf{X}$ is $N \times P$ data matrix, F is factor matrix, L is loading matrix and $E_{.i} \sim N(0, \Psi_{P\times P})$. For individual i, the data $X_{i.}$ can be represented as a linear combina-

tion of latent factors, $F_{.1}, \cdots, F_{.K}$:

$$\begin{aligned} X_{i1} &= F_{11}L_{i1} + F_{21}L_{i2} + \cdots + F_{K1}L_{iK} + E_{i1} \\ X_{i2} &= F_{12}L_{i1} + F_{22}L_{i2} + \cdots + F_{K2}L_{iK} + E_{i2} \\ &\cdots \\ X_{iP} &= F_{1P}L_{i1} + F_{2P}L_{i2} + \cdots + F_{KP}L_{iK} + E_{iP}. \end{aligned}$$

There are methods applying this low rank structure to estimate the covariance matrix and precision matrix. Fan et al. (2008, 2011) applied factor model to estimate the covariance matrix with observed L as follows:

$$\Sigma = F^T cov(L)F + \Psi. \tag{4.3.13}$$

They assume that Σ_E the covariance matrix of error term E, which is diagonal or sparse matrix, and E and L are independent. They applied sample covariance matrix as a estimation of $cov(L)$:

$$\widehat{cov}(L) = \frac{1}{N}\sum_i (L_i - \bar{L})^T (L_i - \bar{L}); \bar{L} = \frac{1}{N}\sum_i L_i.$$

They provided a estimation of precision matrix, which is obtained based on Woodbury's matrix identity. For latent factor model, Fan et al. (2016) applied the same decomposition of the covariance matrix as (4.3.13) and the latent factors need to be estimated.

Following Fan et al. (2011, 2016), we use the same procedure to estimate the

covariance/precision matrix and apply sparse factor analysis to estimate the latent factors. In this chapter, we consider approaches such as FLASH and SFAmix (Gao et al., 2013) for sparse factor analysis. There are many other possible methods we can apply for latent factor model, such as PCA, Robust PCA (Candès et al., 2011), factor analysis (Engelhardt and Stephens, 2010; Gao et al., 2013) and matrix decomposition (Witten et al., 2009; Aharon et al., 2006; Zou et al., 2006b).

Sparse factor analysis

In this chapter, we apply sparse factor analysis to estimate the low rank structure. Factor model with sparsity is desirable in many scenarios which leads to a challenge of how to introduce proper sparsity. Sparsity plays an important role in latent factor models, because it is easier to interpret due to only a small number of variables associated with the latent factors, and has a better performance in prediction due to only keeping representative elements in factors. In practice, not all observed variables should be correlated with latent factors, so introducing sparsity on factors makes practical sense as well. It provides a regularization mechanism to deal with identifiability issues and overfitting problem, and reflects our prior belief on the true nature of the latent structure. The flexibility of sparsity for factors is the key property in this procedure, since we know that there are sparse factors and dense factors in the latent structure. In sparse factor analysis, there are many methods to introduce sparsity on the factors, such as ARD prior (Engelhardt and Stephens, 2010), spike and slab prior (Carvalho et al., 2008), three parameters beta prior (Gao et al., 2013) and adaptive shrinkage prior (chpater 2).

SFAmix (Gao et al., 2013) assumes that the factors comes from mixture of dense and sparse prior which is called three parameters beta (TPB) distribution. The sparsity of the factors is introduced by this TPB distribution. The hierarchical structure of prior (TPB distribution) of factors is represented as follows:

$$\begin{aligned}
\gamma &\sim G(f, \nu) \\
\eta &\sim G(e, \gamma) \\
\tau_k &\sim G(d, \eta) \\
\phi_k &\sim G(c, \tau_k) \\
\delta_{jk} &\sim G(b, \phi_k) \\
\theta_{jk} &\sim G(a, \delta_{jk}) \\
F_{jk} &\sim N(0, \theta_{jk})
\end{aligned}$$

To model both sparse factor and dense factor into this sparse factor analysis, an indicator is introduced:

$$\begin{aligned}
\pi &\sim Beta(\alpha, \beta) \\
Z_k &\sim Bern(\pi) \\
F_{jk}|Z_k &\sim \begin{cases} P(F_{jk}|\theta_{jk}, \delta_{jk}, \phi_k) \\ P(F_{jk}|\phi_k) = N(0, \phi_k) \end{cases}
\end{aligned} \tag{4.3.14}$$

where $P(F_{jk}|\theta_{jk}, \delta_{jk}, \phi_k)$ is three parameters beta distribution as described above and $N(0, \phi_k)$ represent the factor is dense.

FLASH (chapter 2) assumes the prior for L and F are as following distributions:

$$L_{ik} \sim g_l^k(L_{ik}) \tag{4.3.15}$$

$$F_{jk} \sim g_f^k(F_{jk}) \tag{4.3.16}$$

where $g_l^k(\cdot)$ and $g_f^k(\cdot)$ can be any symmetric unimodal distributions. FLASH introduces adaptive shrinkage prior (Stephens, 2016) to both factors and loadings.

Based on the results from sparse factor analysis, the estimation of covariance matrix is:

$$\hat{\Sigma} = \Psi + F^T \Lambda_L F \tag{4.3.17}$$

where $\Lambda_L = \frac{1}{N} L^T L$ or $\Lambda_L = \frac{1}{N} diag(L^T L)$, which is a estimation of covariance matrix of L. It is because we assume that L and F are independent with E and L_i are independent with each other for $i = 1, \cdots, N$ with mean zero. Based on Woodbury's matrix identity, the precision matrix estimation is as follows:

$$\begin{aligned} \hat{\Omega} &= \hat{\Sigma}^{-1} \\ &= \Psi^{-1} - \Psi^{-1} F^T (\Lambda_L^{-1} + F \Psi^{-1} F^T)^{-1} F \Psi^{-1} \end{aligned} \tag{4.3.18}$$

In this subsection, we assume that Ψ is a diagonal matrix. Based this assumption, there are two interesting observations:

- Both $\hat{\Sigma}$ and $\hat{\Omega}$ are diagonal plus low rank.

- Inverse $\hat{\Sigma}$ is not computationally intensive.

The first one is due to the inverse of Ψ is still diagonal and the rank of $\Psi^{-1}F^T(\Lambda_L + F\Psi^{-1}F^T)^{-1}F\Psi^{-1}$ smaller than K. The second one is because Ψ is diagonal, which mean the inverse Ψ^{-1} is easy to calculate, and $(\Lambda_L + F\Psi^{-1}F^T)^{-1}$ just need solve the inverse of a $K \times K$ matrix.

4.3.3 Hybrid Method

In this section, we introduce a hybrid method to combine the two types of previous methods together. The hybrid method incorporate the sparse precision matrix method into the low rank method frame work. This method is based on the conditional sparsity in precision matrix. In the beginning, we would introduce two types of conditional sparsity.

Conditional sparsity

Fan et al. (2016) introduced conditional sparsity (given factors), in which all the variables $X_{\cdot 1}, \cdots X_{\cdot P}$ are still mutually correlated given the factors, though the mutual correlations are weak. It is reasonable to relax the diagonal assumption of Ψ to non-diagonal or sparse. If the factor is given or known, the estimating procedure is:

1. Get the loadings by solving a regression problem given factors.

2. Calculate the sample covariance matrix of the residuals matrix.

3. Apply any threshold method on the elements of covariance matrix of residuals.

After this procedure, we need ensure $\hat{\Psi}$ to be positive definite. One possible way is to control the smallest eigen value by setting proper tuning parameter. Fan et al. (2016) introduced ways to make it positive definite based on works from Qi and Sun (2006); Zou et al. (2014) and R-package "nearPD" (Higham, 2002).

In this section, we consider another conditional sparsity of $\mathbf{X}$ given the factors. This is different from the conditional sparsity introduced in Fan et al. (2016), which assumes that mutually correlations of $X_{.1}, \cdots X_{.P}$ are sparse. This type of conditional sparsity is the assumption that precision matrix of error term E, Ψ^{-1}, is sparse. Based on the model (4.3.12), the conditional sparsity can be interpreted as given the latent factors, $f_1, \cdots, f_K$, and loadings, $l_1, \cdots, l_K$, the conditional correlations (partial correlation) of $X_{.1}, \cdots X_{.P}$ are sparse. Thus, we denote the first type of conditional sparsity as conditional sparsity on correlation (conditional sparsity on covariance matrix) and the second type of conditional sparsity as conditional sparsity on partial correlation (conditional sparsity on precision matrix).

We can deal with conditional sparsity on covariance matrix by using sparse factor analysis and allowing the rank K is large enough. In the conditional sparsity on covariance matrix, the covariance matrix can be decompose as a low rank matrix and a full rank sparse matrix as 4.3.17. We can write the factor model as follows:

$$\mathbf{X}_{N\times P} = L'_{N\times K_1} F'_{K_1\times P} + L''_{N\times K_2} F''_{K_2\times P} + E_{N\times P} \tag{4.3.19}$$

where K_1 is much smaller than $\min(N, P)$ and K_2 is larger than K_1, which can be as big as $\max(N, P)$, and L'', F'' is very sparse. And we still assume that $E_i \sim N(0, \Psi)$ with Ψ being diagonal. In this case, we assume that L', L'', F', F'' are mutually

uncorrelated. The decomposition 4.3.17 can be rewritten as:

$$\hat{\Sigma} = F'^T \Lambda_{L'} F' + F''^T \Lambda_{L''} F'' + \Psi \tag{4.3.20}$$

where $F''^T \Lambda_{L''} F'' + \Psi$ is sparse and full rank since Ψ is diagonal and $F''^T \Lambda_{L''} F''$ is sparse. Thus the decomposition is still a low rank matrix plus a full rank sparse matrix. This property means that by applying sparse factor analysis with assumption of Ψ being diagonal, we can deal with conditional sparsity on covariance matrix.

The conditional sparsity on precision matrix assumes $E_i \sim N(0, \Psi)$ with off-diagonal elements of Ψ^{-1} being sparse. Since the main purpose of this paper is to estimate the precision matrix of $\mathbf{X}$. Based on 4.3.18, the structure of precision matrix $\mathbf{X}$, Ω, is a sparse matrix plus a low rank matrix:

$$\hat{\Omega} = \Psi^{-1}_{P\times P} - \Psi^{-1}_{P\times P} F^T_{P\times K} (\Lambda_L^{-1} + F\Psi^{-1}F^T)^{-1}_{K\times K} F_{K\times P} \Psi^{-1}_{P\times P}$$

where Ψ^{-1} is sparse and $\Psi^{-1}F^T(\Lambda_L^{-1} + F\Psi^{-1}F^T)^{-1}F\Psi^{-1}$ is low rank matrix.

This structure is reasonable in practice. If (X_O, X_H) is joint normal with sparse precision matrix, it does not imply that the precision matrix for X_O is sparse, where we denote X_O as the observed variables and X_H as unobserved (hidden) variables. That is, the appropriateness of the sparsity assumption may depend on which variables in a system have been observed, and is not robust to some variables being unobserved. Given the joint precision matrix is sparse, the marginal precision ma-

trix of X_O is:

$$(\Sigma_{OO})^{-1} = \Omega_{OO} - \Omega_{OH}\Omega_{HH}^{-1}\Omega_{HO} \tag{4.3.21}$$

(Chandrasekaran et al., 2012), where

$$\begin{pmatrix} X_O \\ X_H \end{pmatrix} \sim N\left(0, \begin{pmatrix} \Sigma_{OO} & \Sigma_{OH} \\ \Sigma_{HO} & \Sigma_{HH} \end{pmatrix}\right)$$

and

$$\begin{pmatrix} \Omega_{OO} & \Omega_{OH} \\ \Omega_{HO} & \Omega_{HH} \end{pmatrix} \begin{pmatrix} \Sigma_{OO} & \Sigma_{OH} \\ \Sigma_{HO} & \Sigma_{HH} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$$

In 4.3.21, $(\Sigma_{OO})^{-1}$ is summation of a sparse matrix and a low rank matrix.

To deal with conditional sparsity on precision matrix, we introduce a hybrid method combine sparse factor analysis and sparse precision matrix estimation. The algorithm of this method as follows:

Algorithm 5 Hybrid Method

1: **procedure** SPARSE + LOW-RANK ESTIMATION

2: Get estimated $\hat{L}, \hat{F}$ in model $\mathbf{X} = LF^T + E$

3: Find a sparse estimation of the precision matrix of the residual $\hat{E}$, $\hat{\Psi}^{-1}$.

4: The plug-in estimation: $\hat{\Omega} = \hat{\Psi}^{-1} - \hat{\Psi}^{-1}\hat{F}^T(\Lambda_L + \hat{F}\hat{\Psi}^{-1}\hat{F}^T)^{-1}\hat{F}\hat{\Psi}^{-1}$

5: **return** $\hat{\Omega}$

For the step 1 in Algorithm 5, we can choose any sparse factor analysis or matrix

decomposition method, such SFAmix, FLASH, PMD and PCA, to get the estimation $\hat{L}$ and $\hat{F}$. In the step 2, we can construct the estimated residual matrix as $\hat{E} = X - \hat{F}\hat{L}$, and apply the methods mentioned in sparse precision matrix estimation section, such as GLASSO, CLIME and TIGER. The computational complexity in step 3 is just matrix multiplication of $P \times P$ matrix and $P \times K$ matrix.

The challenge in conditional sparsity on covariance matrix is to make $\hat{\Sigma}$ positive definite. In conditional sparsity on precision matrix, we also need to make $\hat{\Omega}$ symmetric and positive definite. The symmetric and positive definite properties of $\hat{\Psi}^{-1}$ can fit this requirement. GLASSO and other methods can achieve this when Ψ^{-1} is truly sparse.

4.3.4 Assessment of performance

To make comparison and assessment of the performance among different methods of estimating the precision matrix, we provide three different score functions. In the numerical studies, we split the data into two part, training data and test data. We apply precision matrix estimation of the training data set to get $\hat{\Omega}$, and use this estimation and the test date set to evaluation the performance of these methods. If we know the the true precision matrix, we can directly compare the estimation against the truth by using $||\hat{\Omega} - \Omega||_F^2$, where Ω is the true value of the precision matrix. However in practice, we can't observe the precision matrix, so the score function should not depend on the true value. We try to use test data to evaluate the performance of the estimation. In this section we provide three different score functions: log-likelihood, Hyvarinen score and prediction error.

Log-likelihood

This score function is based on the negative log-likelihood function of the test data set given the Ω equal to the estimation $\hat{\Omega}$ from the training data set. The log-likelihood of the test data is:

$$\begin{aligned} & \sum_{i=1}^{N} -\frac{P}{2}\log(2\pi) + \frac{1}{2}\log|\Omega| - \frac{1}{2}\mathbf{tr}(X_i^T \Omega X_i) \\ = & -\frac{NP}{2}\log(2\pi) + \frac{N}{2}\log|\Omega| - \sum_i \frac{1}{2}\mathbf{tr}(\Omega X_i X_i^T) \\ = & -\frac{NP}{2}\log(2\pi) + \frac{N}{2}\log|\Omega| - \frac{N}{2}\mathbf{tr}(\Omega S) \end{aligned}$$

where $S = \frac{1}{N}\mathbf{X}^T\mathbf{X} = \frac{1}{N}\sum_i X_i X_i^T$ assuming $\mathbf{X}$ is centered. We use the negative of the log-likelihood function as score function.

$$L(\hat{\Omega}, \mathbf{X}) = \log|\hat{\Omega}| - \mathbf{tr}(S\hat{\Omega}) \tag{4.3.22}$$

where $S = \frac{1}{N}\mathbf{X}^T\mathbf{X}$ is the sample covariance matrix of the test data and $\hat{\Omega}$ is estimated from training data. We can also call this log score since $L(\Omega, \mathbf{X}) = -\log P_\Omega(\mathbf{X}) + C$, where C is constant with respect to Ω. This constant is related with P, so we should compare this score in data sets with same P. In this chapter, we denote $L(\Omega, \mathbf{X})$ as "loglik".

Hyvarinen score

We can also use Hyvarinen score which is:

$$s(\Omega, \mathbf{X}) = \sum_i [\frac{1}{2}||\nabla \log P(X_i; \Omega)||^2 + \sum_j \frac{\partial^2 \log P(X_i; \Omega)}{\partial^2 X_{ij}}] \tag{4.3.23}$$

Hyvarinen score is proper score rule (Hyvarinen 2005, Parry et al., 2012). Under Gaussian assumption, we know that:

$$\begin{aligned} \nabla \log P(X_i; \Omega) &= -\Omega X_i \\ \frac{\partial^2 \log P(X_i; \Omega)}{\partial^2 X_{ij}} &= -\Omega_{jj} \end{aligned}$$

the score function can be written as follows

$$\begin{aligned} s(\Omega, \mathbf{X}) &= \sum_i (\frac{1}{2} X_i^T \Omega \Omega X_i + \sum_j \Omega_{jj}) \\ &= \sum_i (\frac{1}{2} \mathbf{tr}(\Omega X_i X_i^T \Omega) + \mathbf{tr}(\Omega)) \\ &= N(\frac{1}{2} \mathbf{tr}(\Omega S \Omega) - \mathbf{tr}(\Omega)) \end{aligned}$$

Take derivative to the expectation of this score function, $\frac{1}{2}\mathbf{tr}(\Omega\Sigma\Omega) - \mathbf{tr}(\Omega)$, and get the first order condition:

$$\frac{1}{2}\Sigma\Omega + \frac{1}{2}\Omega\Sigma - I = 0$$

We can see that $\Omega = \Sigma^{-1}$ is the solution, so it is a proper score. So the final score

function we use is as follows:

$$L(\hat{\Omega}, \mathbf{X}) = \frac{1}{2}\mathbf{tr}(\hat{\Omega}S\hat{\Omega}) - \mathbf{tr}(\hat{\Omega}) \tag{4.3.24}$$

where S is still the sample covariance matrix of the test data and $\hat{\Omega}$ is estimated from training data.

To get better understanding about this score, we start from Fisher's divergence:

$$D_f(p, q) = E_p[||\nabla \log p(\mathbf{X}) - \nabla \log q(\mathbf{X})||_2^2]$$

which is expected squared distance between the gradients of the log-densities of the two distributions: p and q. Hyvarinen (2005) shows that, under some regularity conditions, the above function can be written as:

$$D_f(p, q) = E_p[||\frac{1}{2}||\nabla \log q(\mathbf{X})||^2 + \Delta \log q(\mathbf{X})||_2^2] + C_0$$

where C_0 is a constant. The kernel term of the expectation is Hyvarinen score, and we substitute the $q(\mathbf{X})$ with $P(\mathbf{X}, \Omega)$ to get the 4.3.23.

Prediction error

This score function is corresponding to the partial correlation based methods which apply regression methods. In each regression problem, the objective function is $||X_i - \sum_{j \neq i} \beta_j^i X_j||_2^2$ for each i. We consider the square loss of each regression problem

and take the total loss of all regression problem by adding up all the loss functions:

$$L(\hat{\Omega}, \mathbf{X}) = \sum_i ||X_i - (\sum_{j \neq i} -\hat{\Omega}_{ij}/\hat{\Omega}_{ii} X_j)||_2^2 \tag{4.3.25}$$

where $X_1, \cdots, X_P$ are columns of test data and $\hat{\Omega}$ is estimated from training data. We can see that this score is related with P, so we should compare this score in data sets with same P. Although it is actually sum of square error, we denote this score as "MSE" in this chapter.

4.4 Numerical Studies

We apply DSC to make comparisons on precision matrix estimation. We consider two different types of situations:

- methods-oriented: comparing performance of different methods on certain types of data sets.
- data-oriented: comparing performance of methods for specific data set.

We use simulated data and real data to illustrate how to apply DSC in these two situations. Besides the estimating methods mentioned in Section 4.3, we can also apply a straightforward method when N is larger than P, which is taking the inverse of the covariance matrix (denoted as InvCov). The methods in the comparison for simulated data and real data are: InvCov (straightforward method); GLASSO (penalize likelihood method); TIGER(partial correlation based methods); CLIME (sparse column method); SFAmix and FLASH (methods based on low rank assumption) and

Table 4.1: Methods in comparison

Methods			
Method Name	Package	Criteria of Tuning Parameters	Reference
InvCov	solve()		
GLASSO	huge	stars or RIC	(Friedman and Tibshirani, 2008)
TIGER	camel	CV	(Liu and Wang, 2017)
CLIME	camel	CV	(Cai et al., 2011)
SFAmix	SFAmix	BCV or none	(Gao et al., 2013)
FLASH	flashr	BCV or none	
HM	flashr+camel	BCV + CV	

HM (hybrid method). For the hybrid method, we use FLASH or SFAmix to get the low rank part and apply TIGER for the sparse part. For the choice of maximum rank for factor analysis, we use $\min(N, P)$ for FLASH and SFAmix and use BCV to estimate the rank in hybrid method. More details of methods are included in Table 4.1.

In Table 4.1, CV stands for cross-validation, BCV stands for Bi-cross-validation, stars is the stability approach for regularization selection (StARS, Liu et al. (2010)), and RIC is a modified rotation information criterion (RIC, Lysen (2009)). For the low rank methods, "none" means that we can set the maximum rank to be $\min(N, P)$. To assess the performance of methods in different cases, we use three types of score functions in Section 4.3.4.

4.4.1 Simulated datasets

In this section, we simulate data sets from multivariate normal distribution given different types of covariance matrix Σ. We consider the following scenarios: diagonal, toeplitz, band graph, cluster graph, dense Edors Renyi graph, sparse Edors Renyi graph, hub graph and scale free graph. In the scenarios of Diagonal and toeplitz, we generate data sets with covariance matrix as diagonal and toeplitz matrix. The toeplitz matrix has the following form:

$$\begin{pmatrix} a & b & c & d & \cdots & 0 \\ b & a & b & c & \cdots & 0 \\ c & b & a & b & \cdots & 0 \\ & \ddots & \ddots & \ddots & \ddots & \\ 0 & \cdots & c & b & a & b \\ 0 & \cdots & d & c & b & a \end{pmatrix}.$$

In other scenarios, we generate the data sets based on the undirected graph. R-package 'huge' (Zhao et al., 2012) provides a function 'huge.generator' to get different patterns of graph with data matrix. Here we show two examples from the package. Examples of other scenarios are provided in Appendix C.1.

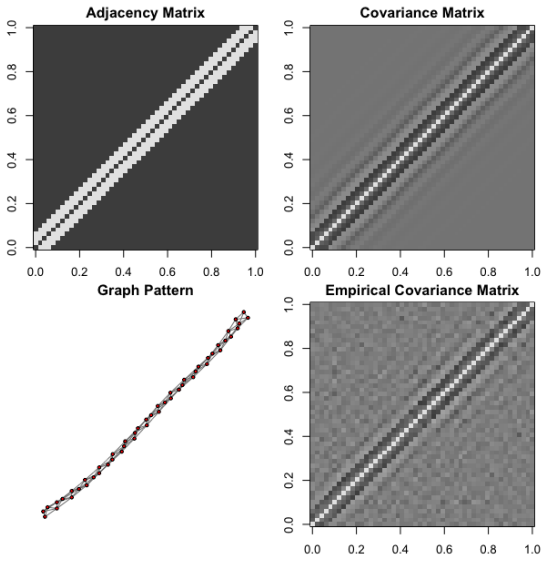

Figure 4.1: Band graph with each node connected with other $K = 3$ nodes.

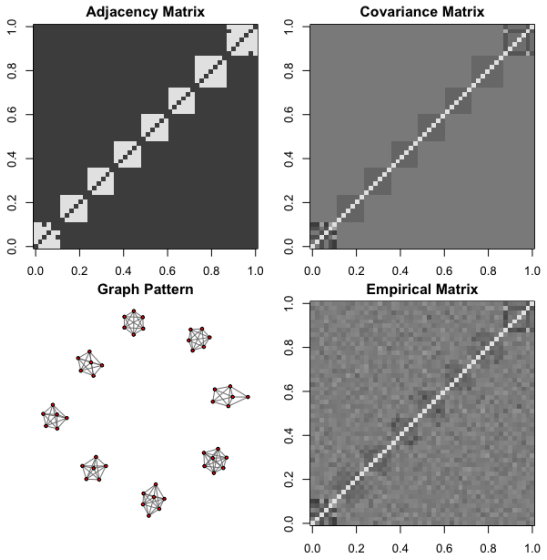

Figure 4.2: Cluster graph with $K = 8$ clusters.

We simulate four sets of data: $N = 200, P = 10$, $N = 200, P = 100$, $N = 200, P = 300$ and $N = 200, P = 1000$ with different types of Σ or Ω. In each simulated data set, we randomly divide it into two parts with equal sample size: training data and test data. We apply different methods on the training data sets to get the estimation of precision matrix. The score functions are calculated using test data and the estimated precision matrix to assess performance. In Figure 4.3-4.6, we show the performance of different methods on three scores. The results are based on 20 random simulations for each cases. Each point in the radar chart is the median value of the scores in a specific scenario. Each axis stands for one specific scenario and the scale ranges from maximum to minimum of the median values from different methods. We want to always keep better scores on the outside of the circle: in loglik score function, where a larger score value means better performance, the score values increases with the distance from the center. In the other two scores, where a smaller score value stands for better performance, larger values stay closer to the center. Thus, the bigger the circle roundness, the better the method performance.

Figure 4.3 shows the results of the scenario where $P = 10$. In this case, $N = 100$ is larger than $P = 10$, so we can directly take the inverse of the sample covariance matrix. For likelihood score, we can see that the InvCov method has better performance than others in most cases other than cluster graph and hug graph, while TIGER works very well in these two cases. TIGER gets better performance than other methods in these three scores over all.

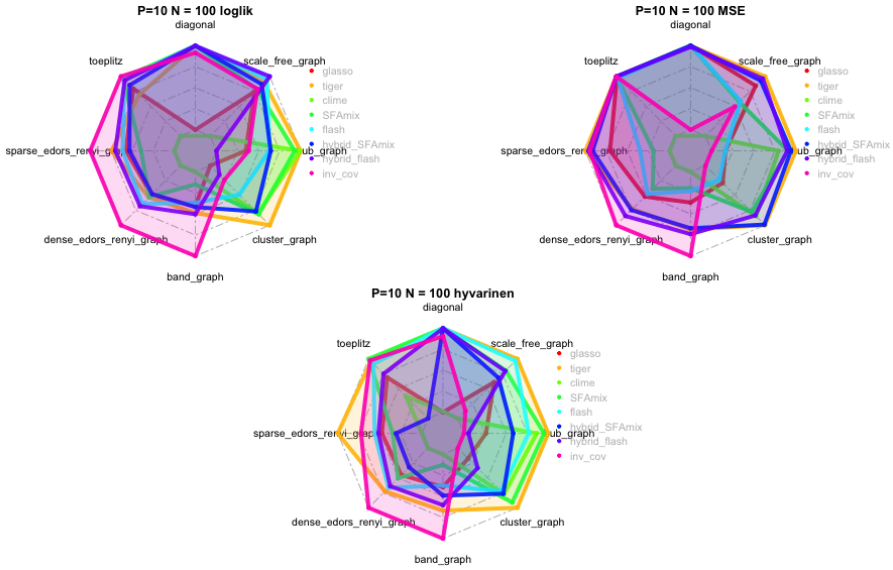

Figure 4.3: Simulated data with $P = 10$.

Figure 4.4-4.6 show the results when P is larger or equal to N. In the comparison of performance among different methods, we can see some pattern from the results. TIGER method has the best performance over all cases among those method based on sparse graph assumption. For loglik score and Hyvarinen, we can see Flash method has better performance in most cases, but it is not as good as hybrid methods in band graph and sparse Edors Renyi graph data sets. Hybrid method with Flash performance reasonably good in most cases comparing with others according to MSE score. For Hyvarinen score, Flash result is no better than that from TIGER method sometimes. From these results, one can tell the strength and weakness of each method on different types of data. Flash is based on the assumption that the data matrix has low rank structure. In cluster graph and hub graph, there are clear low rank structures in data sets, so Flash method works best. In band graph and toeplitz

data, the data matrices are full-rank matrices, which don't fit the assumption of Flash method, so Flash is not as good as TIGER and Hybrid methods. In sparse Edors Renyi graph data, there is no clear low rank structure in data matrix and it fits sparse graph assumption, so TIGER and hybrid methods have the best performance in this case. In general, Hybrid method with Flash has robustly good performance. It works better than TIGER when data has low rank structure, and better than Flash when the data violates the low rank assumption or fits the sparse graph assumption. However there is no methods that beats all others in every situations.

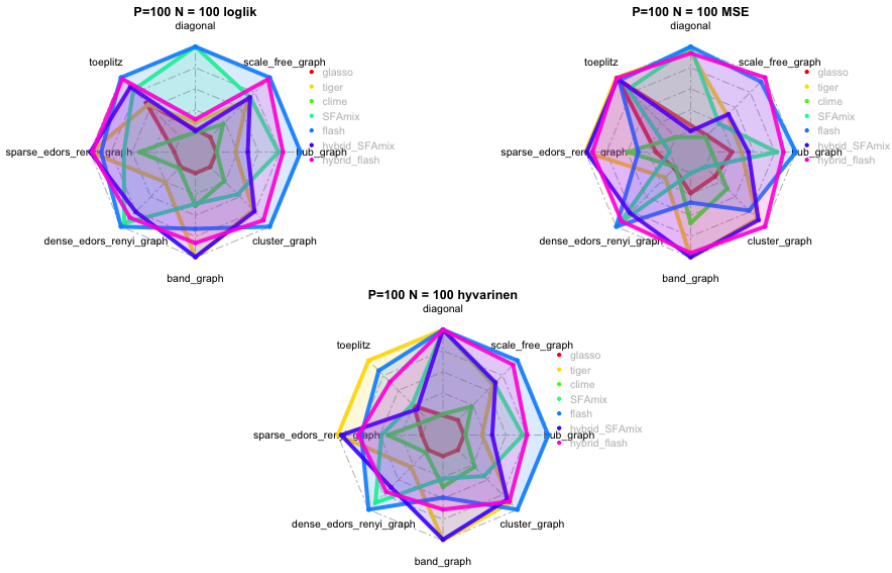

Figure 4.4: Simulated data with $P = 100$.

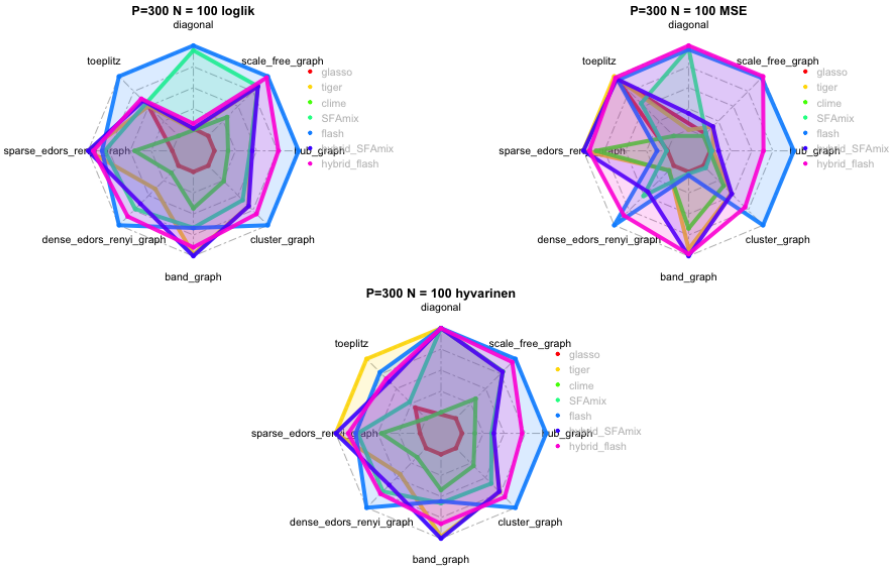

Figure 4.5: Simulated data with $P = 300$.

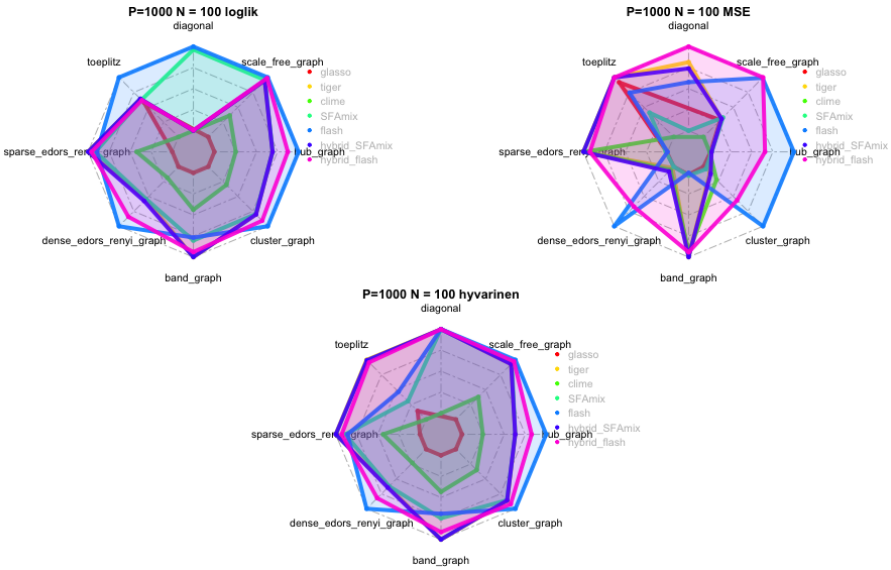

Figure 4.6: Simulated data with $P = 1000$.

4.4.2 Real data analysis

From prior section, we know there is no method outperforms all other methods in precision matrix estimation, and the performance depends on the type of data sets. In practice, it's hard to know whether the data of interest fits certain assumption or not. So we need some data driven suggestions on what method should be applied to specific data. In this section, we focus on the assessment of performance of different methods on one specific real data. We take gene expression data (from GTEx data (Consortium et al., 2015b), Lung tissue) as an example with top 1,000 highly expressed genes, and take quantile normalization on columns of data. Corresponding to sample size and number of variables of the simulated data sets, we randomly draw $N = 100$ samples as training data and $N = 100$ samples as test data without replacement from total 320 samples, and set $P = 10, 100, 300, 1000$. Figure 4.7 shows the results of different methods on three scores based on 20 random draws. In the boxplots of different cases, Flash method and Hybrid method with Flash have better performance than other methods when $P = 100, 300, 1000$. The methods based on sparse graph assumption aren't working very well in this data set. In the case of $P = 10$, we can take the inverse of the sample covariance matrix as the estimation of precision matrix (InvCov), and this method has the best performance among all methods. In this data set, we would suggest apply methods based on low rank assumption to estimate precision matrix for further use, such as multiple testing, classification and other approaches.

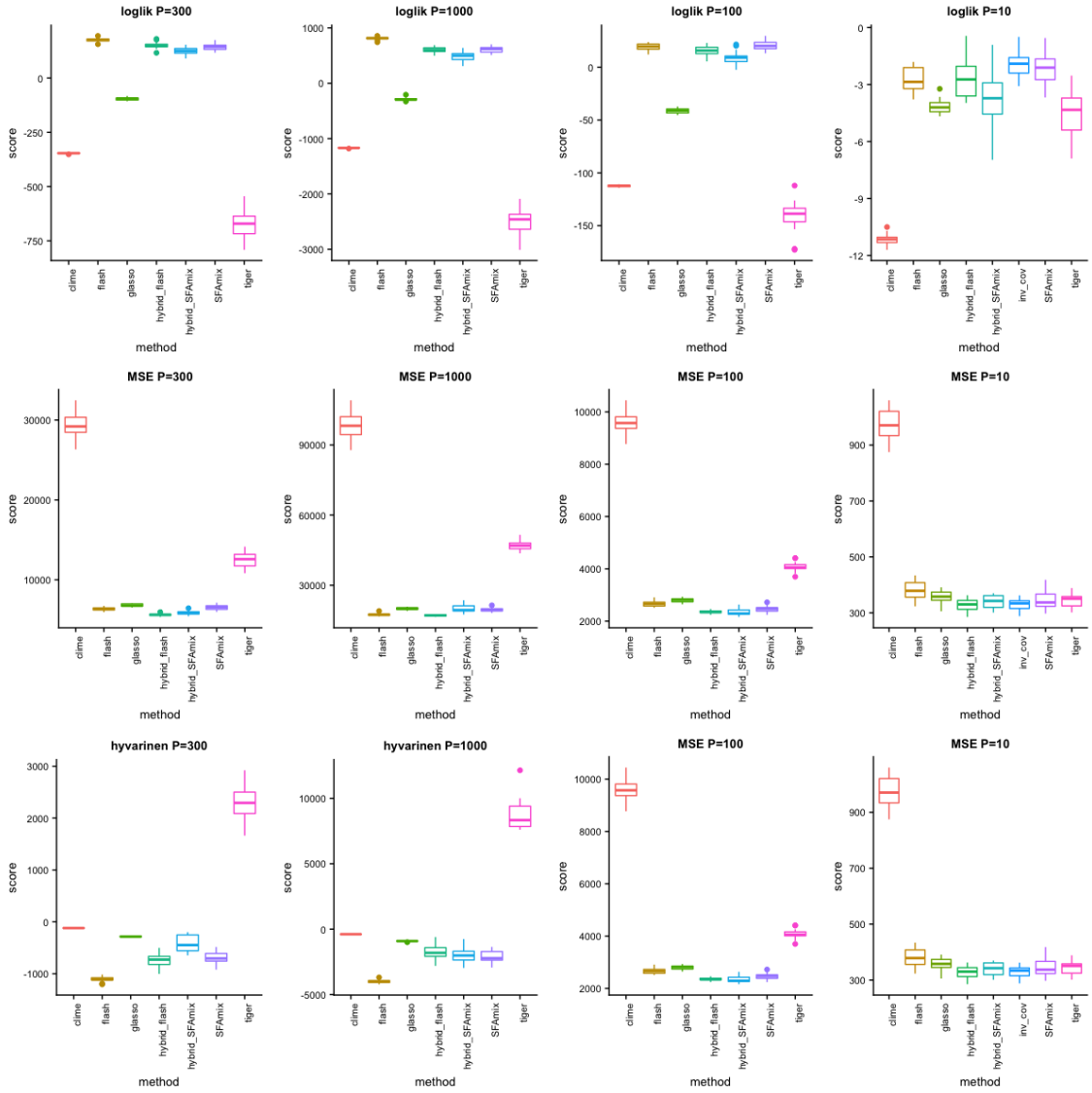

Figure 4.7: Comparison of methods in different cases from Lung data based on three scores.

CHAPTER 5

MR.ASH: MULTIPLE REGRESSION ADAPTIVE SHRINKAGE

5.1 Introduction

Linear regression is a basic problem in statistics, which assumes that the conditional expectation of response, y, given the regressor $\mathbf{X}$, $E(Y|\mathbf{X})$ is a linear combination of variables $X_1, \cdots, X_P$, $E(Y|\mathbf{X}) = \beta_0 + \beta_1 X_1 + \cdots + \beta_P X_P$. Ordinary Least Square (OLS) estimation can be obtained by minimizing the sum of residual square. The OLS estimation is also Maximum Likelihood Estimation (MLE) based on Gaussian assumption. OLS estimation usually has low bias but large variance. To increase the prediction accuracy and control the bias and variance trade-off, we can introduce shrinkage or sparsity on the regression coefficients $\beta_1, \cdots, \beta_P$ to reduce the variance of estimation. We can also get better interpretation by using a small subset of variables to explain the linear correlation between regressors and response. The selection of the best subset of variables is very important problem in linear model, especially for the problem of high-dimensional data.

Variable selection method choosing the optimal model in high dimensional data is a challenge. A variety of methods and algorithms have been proposed to search the optimal model and find the criterion of model selection (Miller, 2002). The best-subset selection is computationally infeasible by seeking the best model over the model space with all possible subsets of variables $(X_1, \cdots, X_P)$. Sparse regression introduce sparsity to coefficients estimation, where the pattern of non-zero entries of

coefficients is corresponding to the model with certain subset of selected variables.

We develop a Bayesian approach for variable selection, and we adapt variational approach from (Carbonetto and Stephens, 2012) for spike and slab prior to a more flexible prior, adaptive shrinkage prior.

5.2 Background: sparse regression

In general, "sparsity" is a powerful idea and especially useful in large-scale data analysis. After introducing sparsity, we can exclude the irrelevant variables into the model to get better predictions and interpretations in high dimensional data, where only a few variables are usually mainly relevant to the response. Sparse estimation avoids overfitting problem, and provides simpler model to understand. Best-subset selection, which can be implemented by the leaps and bounds method (Furnival and Wilson, 1974), is computation-consuming by searching all potential subsets when P is large.

5.2.1 Penalized likelihood methods

Penalized likelihood methods maximize the objective function, $L(\beta; y, \mathbf{X}) - \rho(\beta; \lambda)$, where $L(\beta; y, \mathbf{X})$ is log-likelihood function, $\rho(\beta; \lambda)$ is penalty function and λ is tuning parameter. There are many approaches proposed based on penalized likelihood. Ridge regression (Hoerl and Kennard, 1970) maximizes log-likelihood with L_2 penalty, where all the variables are included into the model with shrinkage. Least absolute shrinkage and selection operator (LASSO), (Tibshirani, 1996), proposed a penalized likelihood method with L_1 penalty, which provides sparse solution by

shrinking some regression coefficients to exact zero. Some limitations of LASSO are pointed in (Zou and Hastie, 2005). For example, LASSO tends to randomly select only one variable from a group of highly correlated variables and the prediction performance is no better than ridge regression with highly correlated variables in the regressors. Ridge regression shrinks the coefficients to each other if the variables are correlated (Friedman et al., 2010). However, the shrinkage of ridge regression gets too large as the proportion of zero coefficients is big, because the shrinkage borrows strengths from each variable. Zou and Hastie (2005) introduced Elastic Net method using a new penalty $\lambda_1||\beta||_1 + \lambda_2||\beta||_2$, where λ_1 and λ_2 are tuning parameters. It works the same as ridge regression when $\lambda_1 = 0$, and it works the same as LASSO when $\lambda_2 = 0$. Elastic Net has better performance in prediction over ridge regression and LASSO in correlated cases (Bhlmann and van de Geer, 2011). There are many other penalized likelihood methods using different penalty terms, such as adaptive LASSO (Zou, 2006b) and Smoothly clipped absolute deviation (SCAD) (Fan and Li, 2001). There are many more methods in the literature such as Yuan and Lin (2006); Efron et al. (2004); Zou (2006a). These methods transform variable selection from discrete parameters to continuous problem, which provides considerable computational convenience. Barber and Candes (2015); Lockhart et al. (2014)provided ways to do further inference.

5.2.2 Bayesian method

In Bayesian framework, sparsity is introduced from the shrinkage prior. This kind of methods can be traced back to methods which propose spike and slab prior on

regression coefficients and applied Markov Chain Monte Carlo algorithms (Mitchell and Beauchamp, 1988; George and McCulloch, 1993, 1997). It's easy to do further inference using the posterior distribution from Bayesian method. There are many approaches for variable selection in Bayesian framework (Hara and Sillanp (2009), a review of Bayesian variable selection methods). Ridge regression can be considered as taking the Maximum a Posteriori (MAP) estimation by using normal prior for β. LASSO can also be interpreted as finding the MAP estimation for linear model assuming regression coefficients from Laplace priors, which is also known as double-exponential prior (Park and Casella, 2008).

Incorporating proper prior information is the key go introduce sparsity. Spike and slab prior is popular in linear model, which puts certain probability on point mass at zero to achieve sparsity and assumes the non-sparse elements follow normal distribution:

$$\beta_j \sim \pi\delta(\beta_j) + (1-\pi)N(\beta_j; 0, \sigma^2). \tag{5.2.1}$$

Spike and slab prior can ensure certain probability at zero in posterior distribution, but it might not perform well in the situation that coefficients are sparse and have heavy tail in distribution. We apply adaptive shrinkage prior introduced in Stephens (2016), and focus on the mixture of normal distribution case:

$$g(\cdot;\pi) \quad = \quad \pi_0\delta_0(\cdot) + \sum_k \pi_k N(\cdot; 0, \sigma_k^2). \tag{5.2.2}$$

This prior is more general than spike and slab prior and has better capability to

capture the tail behavior of the true underlying coefficients distribution.

Given the appealing properties in computational efficiency, we apply variational method to the linear model with *ash* prior to do inference. In Bayesian inference, one possible solution is Markov chain Monte Carlo sampling (Hastings, 1970), in which we sample from a ergodic Markov chain with stationary distribution equal to the posterior. There are different algorithms to do MCMC sampling, such as Metropolis-Hastings algorithm (Hastings, 1970) and Gibbs sampler (Geman and Geman, 1984). In this model, there are too many parameters when P, the number of variables, and K, the number of components of normal mixture, are getting larger. The computational cost of MCMC approach in this case is getting huge when P is large. For high dimensional data, MCMC methods tend to mix slowly because of the multimodal property of the posterior distribution. Variational Inference (Neal and Hinton, 1998; Jordan, 1999; Attias, 1999; Blei et al., 2017) minimizes the Kullback-Leibler (K-L) distance between the approximation and the real posterior density. This optimization procedure also maximizes the lower bound (evidence lower bound) of log likelihood of the data. In this chapter, we can use mean field assumption to make approximated density fully-factorized, which also leads to closed form of the approximated posterior density and makes further statistical inference efficient.

5.3 Model

In this chapter, we focus on Bayesian approach for regression problem. We consider linear model with Adaptive Shrinkage (*ash*) prior, and specifically focus on *ash* prior with mixture of normal components. It can be considered as an extension of varia-

tional inference for Bayesian variable selection in regression ("varbvs") (Carbonetto and Stephens, 2012), which focus on "spike and slab" prior to introduce sparsity. Here we apply a more flexible shrinkage prior to capture the underlying distribution of regression coefficients.

We consider the following linear model:

$$\begin{aligned} y &= \beta_0 \mathbf{1} + \mathbf{X}\beta + \varepsilon \\ &= \beta_0 + X_1\beta_1 + \cdots + X_P\beta_P + \varepsilon \end{aligned} \tag{5.3.1}$$

where $\varepsilon = (\varepsilon_1, \cdots, \varepsilon_n)^T$ and $\varepsilon_i \sim N(0, \sigma_e^2)$ for $i = 1, \cdots, n$, $y = (y_1, \cdots, y_n)^T$ is vector of response of each individual $i = 1, \cdots, n$, $\mathbf{X}$ is $n \times P$ matrix, $\mathbf{X} = (X_1, \cdots, X_P)$, X_j are the covariates $j = 1, \cdots, P$, β_0 is the intercept, $\beta = (\beta_1, \cdots, \beta_P)^T$ is vector of regression coefficients. For each individual i, the response value y_i can be explained by a linear combination of covariates:

$$y_i = \beta_0 + X_{i1}\beta_1 + \cdots + X_{iP}\beta_P + \varepsilon_i. \tag{5.3.2}$$

We assume the prior distribution of regression coefficients, $\beta_1, \cdots, \beta_P$, to be *ash* prior as follows:

$$\beta_j \sim^{iid} \pi_1\delta_0 + \sum_{k=2}^{K} \pi_k N(0, \sigma_e^2\sigma_k^2). \tag{5.3.3}$$

where δ_0 is point mass on zero, $\pi_1 + \cdots + \pi_K = 1$, and σ_e^2 is the variance of the noise

term in model (5.3.1). The variance of each component in the mixture of normal distribution, $(\sigma_2^2, \cdots, \sigma_K^2)$, are pre-fixed.

5.4 Methods

5.4.1 Variational Inference

In variational inference, we try to find a member, q, from a family of density $\mathcal{Q}$, to minimize the Kullback-Leibler (KL) distance between q and the exact posterior density.

$$q^{opt}(\beta) = \arg\min_{\mathcal{Q}} KL(q(\beta)||p(\beta|\mathbf{X}, y)), \tag{5.4.1}$$

$$\begin{aligned} KL(q(\beta)||p(\beta|\mathbf{X}, y)) &= \int q(\beta) \log \left\{ \frac{q(\beta)}{p(\beta|y, \mathbf{X})} \right\} d\beta \\ &= E_q \log q(\beta) - E_q \log p(\beta, y, \mathbf{X}) + \log p(y|\mathbf{X}). \end{aligned} \tag{5.4.2}$$

The solution of this optimization problem is the best approximation of the exact posterior distribution from this family. We can see that KL distance depends on $\log p(y|\mathbf{X})$, which is constant with respect to q and usually hard to compute. We can turn to an equivalent optimization with objective function equal to evidence lower bound (ELBO), where we don't need to calculate $\log p(y|\mathbf{X})$.

$$ELBO(q(\beta)) = E_q \log p(\beta, y|\mathbf{X}) - E_q \log q(\beta). \tag{5.4.3}$$

In this case, we should maximize the $ELBO(q(\beta))$. From (5.4.2), we can see that

$$KL(q(\beta)||p(\beta|\mathbf{X}, y)) + ELBO(q(\beta)) = \log p(y|\mathbf{X}),$$

so $ELBO(q(\beta))$ is a lower bound of log likelihood of the data. In this paper, we use this lower bound as the convergence criteria to get the best approximation.

We focus on the distribution family with mean-field assumption, which assume that β_j are mutually independent:

$$q(\beta) = \prod_{j=1}^{P} q_j(\beta_j). \tag{5.4.4}$$

The performance of this approximate and the complexity of the optimization is determined by the choice of $\mathcal{Q}$. We should choose a flexible density family to capture the exact posterior information (minimize the KL distance) and to make explicit form of the approximation easy to optimize and do inference. In this paper, we focus on the distribution family with the following form:

$$\beta_j \sim \alpha_{j1}\delta_0 + \sum_{k=2}^{K} \alpha_{jk} N(\mu_{jk}, s_{jk})). \tag{5.4.5}$$

We assume that the approximation of the posterior density can be written as a mixture of point mass density and normal densities for each component β_j, where $\alpha_{jk}, \mu_{jk}, s_{jk}; k = 2, \cdots K$ are distinct variational parameters for each j.

5.4.2 Algorithm

Finding the best approximation from $\mathcal{Q}$ becomes a parametric optimization problem for the variational parameters $\alpha_{jk}, \mu_{jk}, s_{jk}$ to minimize the Kullback-Leibler distance. We apply a commonly used algorithm, coordinate ascent, to solve this optimization problem by taking partial derivatives of the KL distance and set the partial derivatives of each variational parameters to zero. More details are included in Appendix D.1. In our regression model,

$$p(\beta, y|\mathbf{X}) = p(y|X, \theta)p(\beta|\theta)$$

so the evidence lower bound is a function of approximation q, and hyper parameters θ, which is denoted as $ELBO(q, \theta)$ with the following analytical expression:

$$\begin{aligned} ELBO(q,\theta) = &-\frac{n}{2}\log(2\pi\sigma_e^2) - \frac{\|y - \mathbf{X}r\|^2}{2\sigma_e^2} \qquad (5.4.6) \\ &- \frac{1}{2\sigma_e^2}\sum_{j=1}^{P}(\mathbf{X}^T\mathbf{X})_{jj}(\sum_{k=2}^{K}\alpha_{jk}(\mu_{jk}^2 + s_{jk}^2) - r_j^2) \\ &- \sum_{j=1}^{P}\sum_{k=1}^{K}\alpha_{jk}\log\left(\frac{\alpha_{jk}}{\pi_k}\right) \\ &+ \sum_{j=1}^{P}\sum_{k=2}^{K}\frac{\alpha_{jk}}{2}\left[1 + \log\left(\frac{s_{jk}^2}{\sigma_e^2\sigma_k^2}\right) - \frac{s_{jk}^2 + \mu_{jk}^2}{\sigma_e^2\sigma_k^2}\right], \qquad (5.4.7) \end{aligned}$$

where $\|\cdot\|$ is the Euclidean norm, r is a column vector with entries $r_j = \sum_{k=2}^{K}\alpha_{jk}\mu_{jk}$. We can also use $ELBO(q, \theta)$ as the objective function to solve the optimization problem. In this case, we repeatedly maximize $ELBO(q, \theta)$ over q and θ. Following this

iterative alternating procedure, we can monotonically increase this lower bound of log likelihood function. To maximize the objective function over q given θ, we can get the update of variational parameters as follows by setting the first order condition of the objective function to zero:

$$
\begin{aligned}
s_{jk} &= \frac{\sigma_e^2}{(\mathbf{X}^T\mathbf{X})_{jj} + \frac{1}{\sigma_k^2}} && (5.4.8)\\
\mu_{jk} &= \frac{s_{jk}}{\sigma_e^2}\left\{(\mathbf{X}^T y)_j - \sum_{t\neq j}(\mathbf{X}^T\mathbf{X})_{jt} r_k\right\} && (5.4.9)\\
\alpha_{jk} &\propto \sqrt{\frac{s_{jk}}{\sigma_k^2\sigma_e^2}}\pi_k \exp\{\frac{1}{2}\frac{\mu_{jk}^2}{s_{jk}}\} && (5.4.10)\\
\alpha_{j1} &\propto \pi_1 && (5.4.11)
\end{aligned}
$$

with constraint $\sum_{t=1}^K \alpha_{jt} = 1$. Thus, we get the best approximation q in $\mathcal{Q}$. Then we need to maximize over θ given current optimized q. In our model, $\sigma_2^2, \cdots, \sigma_K^2$ are fixed, so we only need to find the solution for σ_e^2 and $\pi_1 \cdots, \pi_K$ to maximize $ELBO(q, \theta)$. Following the similar procedure by setting the partial derivative to zero, we can get the update for the hyper parameters:

$$
\sigma_e^2 = \frac{\|y - \mathbf{X}r\|^2 + \sum_j (\mathbf{X}^T\mathbf{X})_{jj} E_q(\beta_j - r_j)^2 + \sum_j \sum_{k=2}^K \alpha_{jk}(s_{jk}^2 + \mu_{jk}^2)/\sigma_k^2}{n + \sum_j \sum_{k=2}^K \alpha_{jk}} \tag{5.4.12}
$$

and

$$
\pi_k \propto \sum_{j=1}^p \alpha_{jk} \tag{5.4.13}
$$

such that $\sum_k \pi_k = 1$.

The *ELBO* is a good lower bound of the marginal likelihood of observed data when KL distance is small enough. It can be used as a model selection criterion (Blei et al., 2017), which has been explored in (Ueda and Ghahramani, 2002; Beal and Ghahramani, 2003). We use this lower bound as the convergence criterion for our variational algorithm, Algorithm 6.

Algorithm 6 Variational Inference

procedure MR.ASH
  initialize $\alpha_{jk}^{(0)}, \mu_{jk}^{(0)}, \pi_k^{(0)}$
  (actually we just need to initialize $r_j^{(0)}$ and $\pi_k^{(0)}$)
  $\tau = 0$
  while $\epsilon > 1e-6$ **do**
    $\tau = \tau + 1$
    for j in $1 : P$ **do**
      update $s_{jk}^{(\tau+1)}$ as (5.4.8)
      update $\mu_{jk}^{(\tau+1)}$ as (5.4.9)
      update $\alpha_{jk}^{(\tau+1)}$ as (5.4.10)-(5.4.11)
    update $\pi_k^{(\tau+1)}$ as (5.4.13)
    update $\sigma_e^{2(\tau+1)}$ as (5.4.12)
    $\epsilon = ELBO(q^{(\tau+1)}, \theta^{(\tau+1)}) - ELBO(q^{(\tau)}, \theta^{(\tau)})$
    ($ELBO(q, \theta)$ is calculated as (5.4.6))
return $q(\beta), \theta$

5.5 Numerical Studies

5.5.1 Simulation with DSC

We add Mr.ash into the DSC example from https://github.com/stephenslab/dsc2-regression, which is based on the simulation study in Zou and Hastie (2005). This is an example to show how to add new method to the existing DSC object. There are four simulation scenarios in this DSC, which are corresponding to the four examples in section 5 in Zou and Hastie (2005), where $Sim1 \cdots Sim4$ are corresponding to $Example.1 \cdots Example.4$ in the simulation study of the paper with same parameter settings, design and sample size. This DSC contains a reproducible comparison among methods, such as Lasso, Ridge regression, naive elastic net and elastic net, on those examples. In this chapter, we focus on the performance of Mr.ash in prediction, so we use MSE of prediction as assessment. We only need add Mr.ash as a new method and we have no need to rerun anything in the existing DSC object. After running this DSC object, we got results from Mr.ash and compare with the existing results from other methods. We use radar chart to visualize the MSE. In Figure 5.1, each axis stands for one situation and the scale is from minimum to maximum of MSE, where maximum values are close to the center. Thus, larger roundness represents better performance. We can see form the results that Mr.ash performs very well over all.

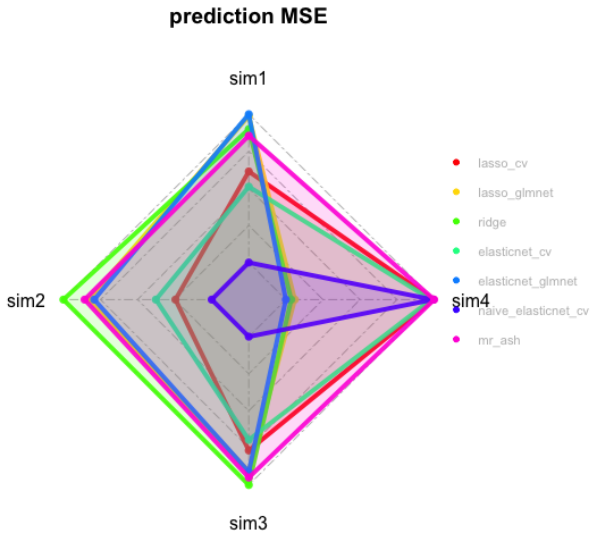

Figure 5.1: The median of prediction MSE is used as score for each method. Each axis is corresponding to one simulation scenario and the scale ranges from maximum to minimum of scores. The maximum is close to the center.

5.6 Related Extension

One straightforward extension of multiple regression with adaptive shrinkage is multiple testing with known covariance structure. Suppose we are interested in detecting the difference between the means of two groups, treatment group and control group, for multiple variables. We can consider it as estimating the difference between the means of these two groups for each variable. In this section, we focus on estimating the "effect" $\beta = (\beta_1, \ldots, \beta_J)$ from observations $\hat{\beta} = (\hat{\beta}_1, \ldots, \hat{\beta}_J)$, where $\hat{\beta}$ are corre-

lated. In multiple testing context, β can be considered as the true difference of the means between two groups and $\hat{\beta}$ can be considered as the observed different of the means. If the "effect" β_j is significantly different from zero, we should reject the null hypothesis which assumes that there is no difference between the means of these two groups for j^{th} variable.

5.6.1 Motivation

The aim of this section is to improve the estimates of β by using "shrinkage" methods. Shrinkage is a powerful idea, but the common problem is how much we need to shrink. Introducing the adaptive shrinkage enables us to learn the amount of shrinkage from the data. Stephens (2016) deals with this problem when $\hat{\beta}_j$ are independent. The method assumes $\hat{\beta}_j|\beta \sim N(\beta_j, s_j)$ with s_j known. In addition, β_j are assumed to be independent and identically distributed from

$$\beta_j \sim g(\cdot)$$

where g can be any unimodal distribution. In this section, we consider symmetric unimodal distribution, since the difference under the null should be centered around zero and symmetric. Corresponding to the prior we choose in previous sections, we choose a specific set of symmetric unimodal distributions, mixture normal distribution,

$$g(\cdot;\pi) = \sum_{k=1}^{K} \pi_k g_k(\cdot;\sigma_k).$$

The component distributions $g_k(\cdot;\sigma_k)$ are normal distribution (with mean 0 and standard deviation σ_k). Here the σ_k are fixed and known, ranging from"small" to "large". The number of components K can be quite large.

Our goal is to adapt *ash* model to allow correlations among observations $\hat{\beta}_j$, specifically $\hat{\beta}|\beta \sim N(\beta, \Sigma)$ where Σ is assumed known. In this case, $\Omega = \Sigma^{-1}$ is also known. In practice, we can't observe Ω, so we assume we have got a reasonable estimator of Ω from any methods. The prior of the "effect" β is still $\beta_j \sim g(\cdot;\pi)$. The main challenges are to estimate π and to obtain the posteriors $\beta_j|\hat{\beta},\hat{\pi}$. We apply variational inference to get the estimation of π and the fully-factorized approximation of posterior distribution of $\beta_j|\hat{\beta},\hat{\pi}$.

5.6.2 Model

As described above, we are interested in estimating the "effect" β_j. Sometimes we can consider the effects being independent as the (Stephens, 2016) is dealing with. This article focuses on tackling with the situation where the effect $\hat{\beta}_j$ are not independent, so we can write our model as follows:

$$\hat{\beta}|\beta, \Sigma \sim N(\beta, \Sigma) \tag{5.6.1}$$

where $\beta = (\beta_1, \cdots, ..., \beta_p)$, $\hat{\beta} = (\hat{\beta}_1, \cdots, \hat{\beta}_p)$ and Σ is the true covariance matrix for $\hat{\beta}$. Here, we assume the estimated effects and the covariance matrix are estimated by any methods. From the inspiration of *ash*, we use the adaptive shrinkage method

on this problem. Then we assume that the effect β_j are independent and identically distributed from some unknown distribution:

$$\beta_j \sim g(.) \tag{5.6.2}$$

We mainly focus on the assumption of unimodal of the prior distribution, therefore we use a mixture of normal distribution in practice, where

$$g(.) = \sum_{k=1}^{K} \pi_k f_k(.) \tag{5.6.3}$$

here we use $f_k(.) = N(.; 0, \sigma_k^2)$.

5.6.3 Inference

To conduct inference on the model above, we can convert this model to regression model and then apply Mr.ash. To rewrite the model (5.6.1), we denote $y = \Sigma^{-\frac{1}{2}}\hat{\beta}$, $\mathbf{X} = \Sigma^{-\frac{1}{2}}$ and $\varepsilon \sim N(0, I)$, then we can construct the regression model as

$$y = \mathbf{X}\beta + \varepsilon \tag{5.6.4}$$

which is the same as $\hat{\beta}|\beta, \Sigma \sim N(\beta, \Sigma)$ by substituting the notations. This is a simplified version of model (5.3.1), where σ_e^2 is known to be one.

This reparameterization help us solve the problem of estimating the "effect" using the update for the parameters of variational inference for multiple regression problem, and then we need to go back to our original problem from this regression framework

to get the update for parameters in our original model structure. In the regression context, all the variational algorithms needed as inputs from the observed data are just $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}^T y$. From our reparameterization, we can obtain $\mathbf{X}^T\mathbf{X} = \Sigma^{-1}$, $\mathbf{X}^T y = \Sigma^{-1}\hat{\beta}$. Denoting $\Omega = \Sigma^{-1}$, all we actually need are $\hat{\beta}$ and Ω which is the inverse of the covariance matrix, so we need a method to estimate the inverse of the covariance matrix. In real data analysis, Ω is not observed and sometimes difficult to be estimated. Thus here we just assume we can get a reasonable estimator for Ω, then apply that as an input of our variational inference to get a fully-factorized approximation of posterior density and the solution of the hyper parameters. Here we still assume that:

$$q(\beta) = \prod_{j=1}^{P} q_j(\beta_j) \tag{5.6.5}$$

and for each β_j:

$$\beta_j \sim \sum_{k=1}^{K} \alpha_{jk} N(\mu_{jk}, s_{jk})). \tag{5.6.6}$$

Following the derivation in 5.4.2 and setting $\sigma_e^2 = 1$, we can get the update for the variational parameters as follows:

$$s_{jk} = \frac{1}{(\Omega)_{jj} + \frac{1}{\sigma_k^2}} \tag{5.6.7}$$

$$\mu_{jk} = s_{jk}((\Omega\hat{\beta})_j - \sum_{t \neq j} (\Omega)_{jt} r_t) \tag{5.6.8}$$

$$\alpha_{jk} \propto \sqrt{\frac{s_{jk}}{\sigma_k^2}} \pi_k exp\{\frac{1}{2}\frac{\mu_{jk}^2}{s_{jk}}\} \tag{5.6.9}$$

$$\sum_{k=1}^{K} \alpha_{jk} = 1 \tag{5.6.10}$$

and the update of hyper parameter π is same as (5.4.13). We use the iterative updating to get the estimation of each parameter, and we use $ELBO$ as the convergence criterion:

$$logP(y|X,\pi) \geq ELBO(\pi, q) = \int q(\beta) log\{\frac{P(y,\beta|\mathbf{X},\pi)}{q(\beta)}\} d\beta.$$

The details of the variational algorithm is the same as Algorithm 6 with setting $\sigma_e^2 = 1$ through the procedure.

5.6.4 Numerical studies

Simulation with known Ω

We compare MVASH and *ash* in simulated data sets with known covariance. We simulate data from multivariate normal distribution $\hat{\beta} \sim N_P(\beta, \Sigma)$, where $P = 1000$ and Σ is known. We generate four data sets with different covariance matrix.

- Diagonal: Σ is a diagonal matrix and $\Sigma_{ii} = 1$.
- Exp decay: $\Sigma_{ii} = 1$, $\Sigma_{ij} = \rho^{|i-j|}$ if $|i-j| \leq K$, where $K = 10, \rho = 0.8$, and $\Sigma_{ij} = 0$ otherwise.
- Toeplitz: $\Sigma_{ii} = 1$, $\Sigma_{ij} = (1 - \frac{|i-j|}{K})^2$ if $|i-j| \leq K$, where $K = 20$, and $\Sigma_{ij} = 0$ otherwise.

- Pairwise: Σ is constructed by blocks of 2×2 matrix. $\Sigma_{ii} = 1$, $\Sigma_{i,(i-(-1)^i)} = \rho$, where $\rho = 0.8$, and $\Sigma_{ij} = 0$ for other (i, j).

For the real effect β, we simulate from mixture of normal distributions and use the same β for all situations in each simulation.

$$\beta_j \sim \pi_0 \delta_0 + \sum_{k=1}^{3} \pi_k N(0, \sigma_k^2),$$

where $(\pi_0, \pi_1, \pi_2, \pi_3) = (0.7, 0.1, 0.1, 0.1)$ and $(\sigma_1, \sigma_2, \sigma_3) = (0.01, 1, 10)$. Figure 5.2 shows the ROC curve for one simulation and Figure 5.3 shows the MSE of β estimation for 100 simulations. We can see that *ash* and MVASH have almost same performance when the covariance is diagonal, which is as we expected. It is because $\hat{\beta}_j$ are independent when Σ is diagonal. MVASH model is same as *ash* model in this scenario. In other scenarios, we can get more information of covariance structure form the off-diagonal terms of Σ, so MVASH have better performance.

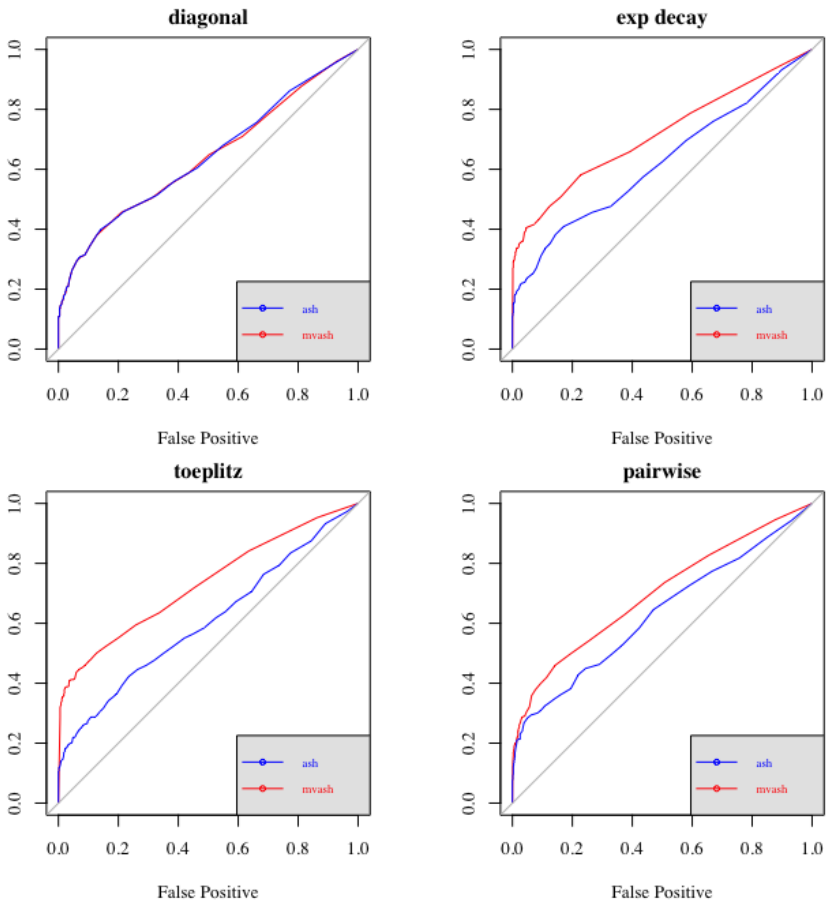

Figure 5.2: True positive rate against false positive rate for different situations in the simulated data set.

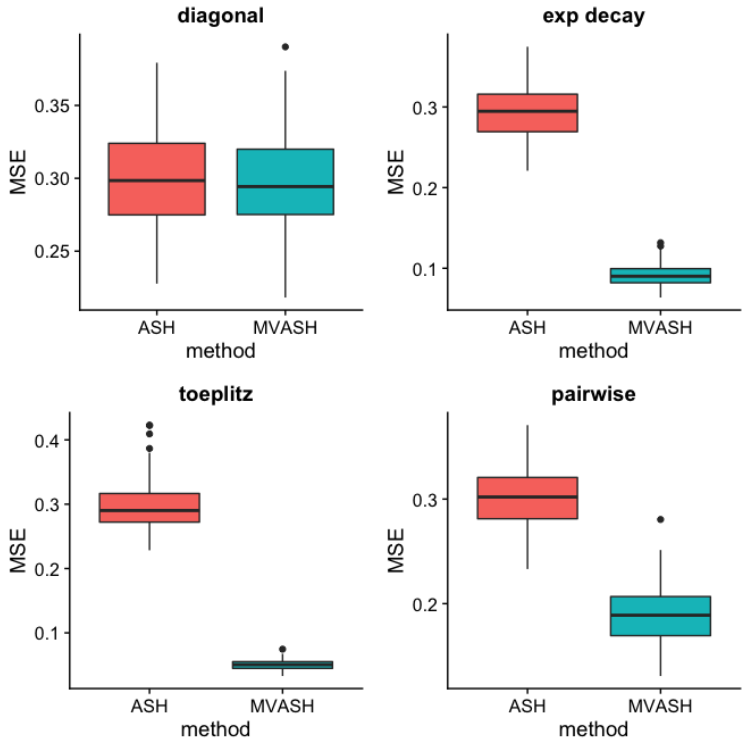

Figure 5.3: MSE of beta estimation for different situations in 100 simulated data sets.

Simulation with estimated Ω from real data

In practice, we can't observe the Σ or Ω. We need a reasonable estimation of Ω, $\hat{\Omega}$. More details about choosing reasonable estimation will be introduced in DSC-Omega. Here, we apply two different methods: Glasso and SFAmix. These two methods are based on different assumption about Ω. Glasso assumes that Ω is sparse and SFAmix assumes that Ω is equal to a diagonal matrix plus a low rank matrix. SFAmix is a method for sparse factor analysis, and we use it to estimate

the low rank matrix. So we call the method to estimate the precision matrix using SFAmix as SFAmix for convenience. In this method, we use sparse factor analysis to estimate the low rank structure of data matrix and use that to get estimation of covariance matrix. We further apply Woodbury's matrix identity to calculate the inverse of covariance matrix. In this experiment, we use a gene expression data with 1,500 genes and 320 samples. After normal quantilization by columns, we randomly split the data into two data sets with equal size. We can call this two data sets as treatment group and control group. And then we add "artificial" effect β to treatment group, where $\beta_j \sim \pi_0\delta_0 + \sum_{k=1}^{2} \pi_k N(0, \sigma_k^2)$, $(\pi_0, \pi_1, \pi_2) = (0.8, 0.1, 0.1)$ and $(\sigma_1, \sigma_2) = (0.05, 0.2)$. The "artificial" gene expression data is prepared in this way. We would like to find which gene is differentially expressed between treatment group and control group. We know that the mean difference follows multivariate normal distribution, $\bar{X}_1 - \bar{X}_2 \sim N(\beta, \frac{n_1+n_2}{n_1 n_2}\Sigma)$ and $n_1 = n_2 = \frac{n}{2} = 160$. In this case, we know the true effect β but covariance structure is unknown, which is from real data. The methods mentioned above are applied to find the estimation, $\hat{\Omega}$. Figure 5.4 shows the ROC curves from *ash* and MVASH. In MVASH, we use different $\hat{\Omega}$ from SFAmix and Glasso with different tuning parameters. There is a tuning parameter in Glasso which control the sparsity of the estimated precision matrix, $\hat{\Omega}$. We use the R package "glasso" in this part. As the tuning parameter getting small, the estimated precision matrix gets dense. The performance depends on the choice of tuning parameter. With proper estimation of precision matrix, MVASH has better performance than *ash* in this multiple testing approach.

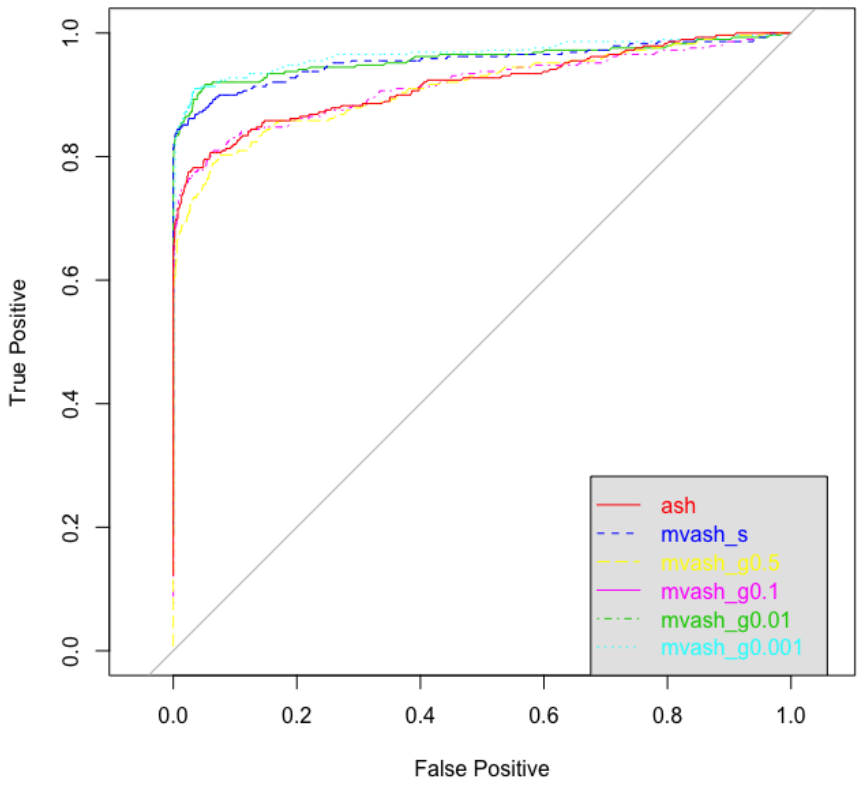

Figure 5.4: ROC curve for different methods. "mvashg" means MVASH using $\hat{\Omega}$ from Glasso with different tuning parameters and "mvashs" means MVASH using $\hat{\Omega}$ from SFAmix.

REFERENCES

Adragni, K. and Cook, D. (2009). Sufficient dimension reduction and prediction in regression. *Philosophical Transactions of the Royal Society A*, 367:4385–4405.

Aharon, M., Elad, M., and Bruckstein, A. M. (2006). The {K-SVD}: An Algorithm for Designing of Overcomplete Dictionaries for Sparse Representations. *IEEE Transactions on Signal Processing*, 54(11):4311–4322.

Attias, H. (1999). A variational bayesian framework for graphical models. *Proceedings of the 12th International Conference on Neural Information Processing Systems*, pages 209–215.

Bai, J. and Ng, S. (2008). Large Dimensional Factor Analysis. *Foundations and Trends® in Econometrics*, 3(2):89–163.

Banerjee, O., Ghaoui, L. E., and dAspremont, A. (2008). Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data. *Journal of Machine Learning Research*, 9:485–516.

Barber, R. F. and Candes, E. (2015). Controlling the false discovery rate via knockoffs. *Annals of statistics*, 43(5):2055–2085.

Beal, M. J. and Ghahramani, Z. (2003). The Variational Bayesian EM Algorithm for Incomplete Data : with Application to Scoring Graphical Model Structures. *Bayesian Statistics*.

Belloni, A., Chernozhukov, V., and Wang, L. (2011). Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806.

Bhaskar, S. A. and Javanmard, A. (2015). 1-Bit Matrix Completion under Exact Low-Rank Constraint. (2).

Bhlmann, P. and van de Geer, S. (2011). *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer Publishing Company, Incorporated, 1st edition.

Bishop., C. M. (2006). Pattern Recognition and Machine Learning. *Springer*.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2016). Variational Inference: A Review for Statisticians.

Blei, D. M., Kucukelbir, A., and Mcauliffe, J. D. (2017). Variational Inference : A Review for Statisticians. pages 1–41.

Breiman, L. and Friedman, J. H. (1985). Estimating Optimal Transformations for Multiple Regression and Correlation. *Journal of the American Statistical Association*, 80(391):580–598.

Cai, T., Liu, W., and Luo, X. (2011). A Constrained 1 Minimization Approach to Sparse Precision Matrix Estimation. *Journal of the American Statistical Association*, 106(494):594–607.

Cai, T. and Zhou, W. (2013). A Max-Norm Constrained Minimization Approach to 1-Bit Matrix Completion. *Journal of Machine Learning Research*, 14:3619–3647.

Candès, E. J., Li, X., Ma, Y., and Wright, J. (2011). Robust principal component analysis? *Journal of the ACM*, 58(3):11:1–11:37.

Candès, E. J., Sing-Long, C., and Trzasko, J. D. (2013). Unbiased risk estimates for singular value thresholding and spectral estimators. *IEEE Trans. Signal Processing*, 61(19):4643–4657.

Carbonetto, P. and Stephens, M. (2012). Scalable Variational Inference for Bayesian Variable Selection in Regression , and Its Accuracy in Genetic Association Studies. *Bayesian Analysis*, (1):73–108.

Carvalho, C. M., Chang, J., Lucas, J. E., Nevins, J. R., Wang, Q., and West, M. (2008). High-Dimensional Sparse Factor Modeling: Applications in Gene Expression Genomics. *Journal of the American Statistical Association*, 103(484):1438–1456.

Chandrasekaran, V., Parrilo, P. A., and Willsky, A. S. (2012). Latent variable graphical model selection via convex optimization. *Annals of statistics*, 40(4):1953–1967.

Clyde, M. and George, E. I. (2000). Flexible Empirical Bayes Estimation for Wavelets. *Journal of the Royal Statistical Society Series B*, 62(4):681–698.

Collins, M., Dasgupta, S., and Schapire, R. E. (2001). A generalization of principal component analysis to the exponential family. *Proc. Adv. in Neural Processing Systems (NIPS)*.

Consortium, . G. P. et al. (2015a). A global reference for human genetic variation. *Nature*, 526(7571):68.

Consortium, G. et al. (2015b). The genotype-tissue expression (gtex) pilot analysis: Multitissue gene regulation in humans. *Science*, 348(6235):648–660.

Cunningham, J. P. and Ghahramani, Z. (2015). Linear Dimensionality Reduction : Survey , Insights , and Generalizations. *Journal of Machine Learning Research*, 16:2859–2900.

Davenport, M. A., Plan, Y., van den Berg, E., and Wootters, M. (2014). 1-Bit Matrix Completion. *Information and Inference: A Journal of the IMA*, 3:189–223.

de Leeuw, J. (2006). Principal component analysis of binary data by iterated singular value decomposition. *Computational Statistics & Data Analysis*, 50(1):21–39.

Drton, M. and Perlman, M. D. (2007). Multiple Testing and Error Control in Gaussian Graphical Model Selection. *Statistical Science*, 22(3):430–449.

Efron, B. (1982). *The jackknife, the bootstrap and other resampling plans*. SIAM.

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

Efron, B. and Morris, C. (1973). Stein's estimation rule and its competitors?an empirical bayes approach. *Journal of the American Statistical Association*, 68(341):117–130.

Engelhardt, B. E. and Stephens, M. (2010). Analysis of population structure: a unifying framework and novel methods based on sparse factor analysis. *PLoS genetics*, 6(9):e1001117.

Fan, J., Fan, Y., and Lv, J. (2008). High dimensional covariance matrix estimation using a factor model. *Journal of Econometrics*, 147(1):186–197.

Fan, J., Feng, Y., and Wu, Y. (2009). Network exploration via the adaptive lasso and SCAD penalties. 3(2):521–541.

Fan, J. and Li, R. (2001). Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties. *Journal of the American Statistical Association*, 96(456):1348–1360.

Fan, J., Liao, Y., and Liu, H. (2016). An overview of the estimation of large covariance and precision matrices. *The Econometrics Journal*, 19(1).

Fan, J., Liao, Y., and Mincheva, M. (2011). High dimensional covariance matrix estimation in approximate factor models. *Annals of statistics*, 39(6):3320.

Foygel, R. and Drton, M. (2010). Extended bayesian information criteria for gaussian graphical models. *Advances in Neural Information Processing Systems 23*.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1):1–22.

Friedman, J. and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9:432–441.

Furnival, G. M. and Wilson, R. W. (1974). Regressions by leaps and bounds. *Technometrics*, 16(4):499–511.

Gao, C., Brown, C. D., and Engelhardt, B. E. (2013). A latent factor model with a mixture of sparse and dense factors to model gene gene expression data with confounding effects. *arXiv:1310.4792v1*, pages 1–28.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

George, E. and Foster, D. P. (2000). Calibration and empirical bayes variable selection. *Biometrika*, 87(4):731–747.

George, E. and McCulloch, R. (1993). Variable Selection via Gibbs Sampling. *Journal of the American Statistical Association*, 88(423):881–889.

George, E. and McCulloch, R. (1997). Approaches for Bayesian variable selection. *Statistica Sinica*, 7:339?373.

Hao, W., Song, M., and Storey, J. D. (2016). Genetics and population analysis Probabilistic models of genetic variation in structured populations applied to global human studies. *Bioinformatics*, 32(November 2015):713–721.

Hara, R. B. O. and Sillanp, M. J. (2009). A Review of Bayesian Variable Selection Methods : What , How and Which. *Bayesian Analysis*, (1):85–118.

Harper, F. M. and Konstan, J. A. (2016). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19.

Hartemink, A. J. (2005). Reverse engineering gene regulatory networks. *Nature Biotechnology*, 23(5):554–555.

Hastings, W. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109.

Higham, N. (2002). Computing the nearest correlation matrix - a problem from finance. *IMA Journal of Numerical Analysis*, 22:329–343.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67.

Hore, V., Viñuela, A., Buil, A., Knight, J., McCarthy, M. I., Small, K., and Marchini, J. (2016). Tensor decomposition for multiple-tissue gene expression experiments. *Nature genetics*, 48(9):1094–1100.

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

Huang, S., Jin, J., and Zhigang, Y. (2016). Partial correlation screening for estimating large precision matrices, with applications to classification. *The Annals of Statistics*, 44:2018–2057.

Hyvarinen, A. (2005). Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research*, 6:695–709.

Hyvarinen, A., Karhunen, J., and Oja, E. (2001). *Independent Component Analysis*. John Wiley and Sons.

Jaakkola, T. S. and Jordan, M. I. (2000). Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10:25–37.

James, W. and Stein, C. (1961). Estimation with quadratic loss. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 361–379.

Johnstone, I. M. and Lu, A. Y. (2004). Sparse principal components analysis. *Unpublished manuscript*, pages 1–29.

Johnstone, I. M., Silverman, B. W., et al. (2004). Needles and straw in haystacks: Empirical bayes estimates of possibly sparse sequences. *The Annals of Statistics*, 32(4):1594–1649.

Jordan, M. I. (1999). An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37:183–233.

Josse, J., Sardy, S., and Wager, S. (2016). denoiseR: A Package for Low Rank Matrix Estimation. (3):1–18.

Klami, A. (2014). P'olya-Gamma augmentations for factor models. *JMLR: Workshop and Conference Proceedings*, 39:112–128.

Knowles, D. and Ghahramani, Z. (2011). Nonparametric Bayesian Sparse Factor. *Annals of Applied Statistics*, 5(2B):1534–1552.

Koenker, R. and Mizera, I. (2010). Quasi-concave density estimation. *The Annals of Statistics*, pages 2998–3027.

Lam, C. and Fan, J. (2009). Sparsistency and rates of convergence in large covariance matrix estimation 1. *The Annals of Statistics*, 37(6):4254–4278.

Lam, C., Yao, Q., and Bathia, N. (2011). *Factor Modeling for High Dimensional Time Series*, pages 203–207. Physica-Verlag HD, Heidelberg.

Lan, A. S., Waters, A. E., Studer, C., and Baraniuk, R. G. (2014). Sparse Factor Analysis for Learning and Content Analytics. *Journal of Machine Learning Research*, 15:1959–2008.

Lauritzen, S. (1996). *Graphical Models*. Oxford University Press.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, S., Huang, J. Z., and Hu, J. (2010). Sparse logistic principal component analysis for binary data. *The Annals of Applied Statistics*, 4:1579–1601.

Leek, J. T. and Storey, J. D. (2008). A general framework for multiple testing dependence. *Proceedings of the National Academy of Sciences of the United States of America*, 105(48):18718–23.

Lefrançais, E., Ortiz-muñoz, G., Caudrillier, A., Mallavia, B., Liu, F., Sayah, D. M., Thornton, E. E., Headley, M. B., David, T., Coughlin, S. R., Krummel, M. F., Leavitt, A. D., Passegué, E., and Looney, M. R. (2017). The lung is a site of platelet biogenesis and a reservoir for haematopoietic progenitors. *Nature Publishing Group*, 544(7648):105–109.

Liu, H., Roeder, K., and Wasserman, L. (2010). Stability Approach to Regularization Selection (StARS) for High Dimensional Graphical Models. *Proceedings of the Twenty-Third Annual Conference on Neural Information Processing Systems*.

Liu, H. and Wang, L. (2017). TIGER : A tuning-insensitive approach for optimally estimating Gaussian. *Electronic Journal of Statistics*, 11:241–294.

Liu, W. and Luo, X. (2015). Fast and adaptive sparse precision matrix estimation in high dimensions. *Journal of Multivariate Analysis*, 135:153–162.

Lockhart, R., Taylor, J., Tibshirani, R. J., and Tibshirani, R. (2014). A significance test for the lasso. *Annals of statistics*, 42(2):413–468.

Lysen, S. (2009). Permuted Inclusion Criterion: A Variable Selection Technique. *PhD thesis, University of Pennsylvania*.

Mazumder, R., Hastie, T., and Tibshirani, R. (2010). Spectral Regularization Algorithms for Learning Large Incomplete Matrices. *Jmlr*, 11:2287–2322.

McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models, Second Edition*. Chapman and Hall.

Meinshausen, N. and Bhlmann, P. (2006). WITH THE LASSO. *The Annals of Statistics*, 34(3):1436–1462.

Miller, A. (2002). *Subset selection in regression*. Monographs on statistics and applied probability. Chapman and Hall, 2nd edition.

Mitchell, T. J. and Beauchamp, J. J. (1988). Bayesian Variable Selection in Linear Regression. *Journal of the American Statistical Association*, 83(404):1023–1032.

Neal, R. and Hinton, G. (1998). A view of the EM Algorithm that justifies incremental, sparse, and other variants. *Learning in Graphical Models*, pages 355–368.

Owen, A. and Perry, P. (2009). Bi-cross-validation of the svd and the nonnegative matrix factorization. *The Annals of Applied Statistics*, 3(2):564–594.

Park, T. and Casella, G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association*, 103(482):681–686.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572.

Polson, N. G., Scott, J. G., and Windle, J. (2013). Bayesian inference for logistic models using plyagamma latent variables. *Journal of the American Statistical Association*, 108(504):1339–1349.

Pournara, I. and Wernisch, L. (2007). Factor analysis for gene regulatory networks and transcription factor activity profiles. *BMC bioinformatics*, 8:61.

Price, A. L., N.j.patterson, R.m.plenge, M.e.weinblatt, and N.a.shadick (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nat. Genet*, 38(8):904–909.

Qi, H. and Sun, D. (2006). A quadratically convergent Newton method for computing the nearest correlation matrix. *SIAM journal on matrix analysis and applications*, 28:360–385.

Ravikumar, P., Wainwright, M. J., Raskutti, G., and Yu, B. (2011). High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980.

Rothman, A. J., Bickel, P. J., Levina, E., and Zhu, J. (2008). Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515.

Sabatti, C. and James, G. M. (2005). Bayesian sparse hidden components analysis for transcription regulation networks. *Bioinformatics*, 22(6):739–746.

Sabatti, C. and James, G. M. (2006). Bayesian sparse hidden components analysis for transcription regulation networks. *Bioinformatics*, 22(6):739–746.

Schein, A. I., Saul, L. K., and Ungar, L. H. (2003). A Generalized Linear Model for Principal Component Analysis of Binary Data. *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*.

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

Sharif, B. and Bresler, Y. (2007). Physiologically improved ncat phantom (pincat) enables in-silico study of the effects of beat-to-beat variability on cardiac mr. *In Proceedings of the 15th Annual Meeting of the International Society for Magnetic Resonance in Medicine (ISMRM)*, page 3418.

Shen, X., Pan, W., and Zhu, Y. (2012). Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107(497):223–232.

Spearman, C. (1904). General intelligence, objectively determined and measured. *American Journal of Psychology*, 15:201–293.

Stegle, O., Parts, L., Piipari, M., Winn, J., and Durbin, R. (2012). Using probabilistic estimation of expression residuals (peer) to obtain increased power and interpretability of gene expression analyses. *Nature protocols*, 7(3):500.

Stein, C. et al. (1956). Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the Third Berkeley symposium on mathematical statistics and probability*, volume 1, pages 197–206.

Stephens, M. (2016). False Discovery Rates: A New Deal. *bioRxiv*, page 038216.

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

Tipping, M. and Bishop, C. (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B,*, 61:611–622.

Ueda, N. and Ghahramani, Z. (2002). Bayesian model search for mixture models based on optimizing variational bounds. *Neural Networks*, 15:1223–1241.

van der Maaten, L. and Hinton, G. E. (2008). Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

West, M. (2003). Bayesian factor regression models in the ?large p, small n? paradigm. *Bayesian Statistics 7 - Proceedings of the Seventh Valencia International Meeting*, pages 723–732.

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

Yang, D., Ma, Z., and Buja, A. (2014). A Sparse Singular Value Decomposition Method for High-Dimensional Data Method for High-Dimensional Data. 8600(May 2017).

Yuan, M. (2010). High dimensional inverse covariance matrix estimation via linear programming. *Journal of Machine Learning Research*, 11:2261–2286.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67.

Yuan, M. and Lin, Y. (2007). Model selection and estimation in the gaussian graphical model. *Biometrika*, 94:19–35.

Zhao, T., Liu, H., Roeder, K., Lafferty, J., and Wasserman, L. (2012). The huge package for high-dimensional undirected graph estimation in R. *Journal of Machine Learning Research*, 13:1059–1062.

Zou, H. (2006a). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429.

Zou, H. (2006b). The Adaptive Lasso and Its Oracle Properties. *Journal of the American Statistical Association*, 101(476):1418–1429.

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

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

Zou, H., Hastie, T., and Tibshirani, R. (2006b). Sparse Principal Component Analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286.

Zou, H., Hastie, T., and Tibshirani, R. (2014). Sparse covariance matrix estimation with eigenvalue constraints. *Journal of Computational and Graphical Statistics*, 23(2):439–459.

APPENDIX A

SUPPLEMENTARY FOR FLASH

A.1 Proof of Proposition 1

The objective function can be written as:

$$F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau}) = \int q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f}) \log \frac{p(Y, \boldsymbol{l}, \boldsymbol{f}; g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \boldsymbol{\tau})}{q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f})} \tag{A.1.1}$$

$$= E_{q_{\boldsymbol{l}}} \left[-\frac{1}{2} \sum_i (A_i l_i^2 - 2 B_i l_i) \right] + E_{q_{\boldsymbol{l}}} \log[g_{\boldsymbol{l}}(\boldsymbol{l}) / q_{\boldsymbol{l}}(\boldsymbol{l})] + C_0 \tag{A.1.2}$$

where C_0 is a constant with respect to $q_{\boldsymbol{l}}$ and $g_{\boldsymbol{l}}$, and

$$A_i = \sum_j \tau_{ij} \mathrm{E}_{q_{\boldsymbol{f}}}(f_j^2) \tag{A.1.3}$$

$$B_i = \sum_j \tau_{ij} Y_{ij} \mathrm{E}_{q_{\boldsymbol{f}}}(f_j) \tag{A.1.4}$$

and we abuse notation slightly by using $g_{\boldsymbol{l}}(\boldsymbol{l})$ to denote the product $\prod_i g_{\boldsymbol{l}}(l_i)$. We now show that maximizing this over $q_{\boldsymbol{l}}$ and $g_{\boldsymbol{l}}$ comes down to solving the EBNM problem.

Normal means problem

Recall the normal means problem:

$$\boldsymbol{x} = \boldsymbol{\theta} + \boldsymbol{e} \tag{A.1.5}$$

$$\theta_1, \ldots, \theta_n \sim^{iid} g, \quad g \in \mathcal{G}. \tag{A.1.6}$$

where $e_i \sim N(0, s_i^2)$.

The EB solution to this problem estimates g by maximum likelihood:

$$\hat{g} = \arg\max_{g \in \mathcal{G}} l(g), \tag{A.1.7}$$

where

$$l(g) = \log p(\boldsymbol{x}|g), \tag{A.1.8}$$

and then finds the posterior distribution of $\boldsymbol{\theta}$ given $\hat{g}$:

$$p(\boldsymbol{\theta}|\boldsymbol{x}, \hat{g}) = \prod_j p(\theta_j|\boldsymbol{x}, \hat{g}) \propto \prod_j \hat{g}(\theta_j) p(x_j|\theta_j, s_j). \tag{A.1.9}$$

Lemma 1. *The Empirical Bayes solution to the normal means problem also solves:*

$$\min_{q_\theta, g \in \mathcal{G}} F^{NM}(q_\theta, g) \tag{A.1.10}$$

where

$$F^{NM}(q_\theta, g) = E_{q_\theta}\left[-\frac{1}{2}\sum_j (A_j\theta_j^2 - 2B_j\theta_j)\right] + \sum_j E_{q_\theta}\log g(\theta_j) - E_{q_\theta}\log q_\theta(\boldsymbol{\theta}) + const \tag{A.1.11}$$

with $A_j = 1/s_j^2$ *and* $B_j = x_j/s_j^2$.

Equivalently, (A.1.10)-(A.1.11) *is solved by* $g = \hat{g}$ *in* (A.1.7) *and* $q_\theta = p(\boldsymbol{\theta}|\boldsymbol{x}, \hat{g})$ *in* (A.1.9)*, with* $x_j = B_j/A_j$ *and* $s_j^2 = 1/A_j$.

Proof. The log likelihood can be written as

$$\begin{aligned}
l(g) &:= \log[p(\boldsymbol{x}|g)] && \text{(A.1.12)}\\
&= \log[p(\boldsymbol{x}, \boldsymbol{\theta}|g)/p(\boldsymbol{\theta}|\boldsymbol{x}, g)] && \text{(A.1.13)}\\
&= \int q_\theta(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{x}, \boldsymbol{\theta}|g)}{p(\boldsymbol{\theta}|\boldsymbol{x}, g)} d\boldsymbol{\theta} && \text{(A.1.14)}\\
&= \int q_\theta(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{x}, \boldsymbol{\theta}|g)}{q_\theta(\boldsymbol{\theta})} d\boldsymbol{\theta} + \int q_\theta(\boldsymbol{\theta}) \log \frac{q_\theta(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\boldsymbol{x}, g)} d\boldsymbol{\theta} && \text{(A.1.15)}\\
&= F^{\mathrm{NM}}(q_\theta, g) + D_{KL}(q_\theta || p_{\theta|x,g}) && \text{(A.1.16)}
\end{aligned}$$

where

$$F^{\mathrm{NM}}(q_\theta, g) = \int q_\theta(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{x}, \boldsymbol{\theta}|g)}{q_\theta(\boldsymbol{\theta})}\, d\boldsymbol{\theta} \tag{A.1.17}$$

and

$$D_{KL}(q_\theta || p_{\theta|x,g}) = -\int q_\theta(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta}|\boldsymbol{x}, g)}{q_\theta(\boldsymbol{\theta})}\, d\boldsymbol{\theta} \tag{A.1.18}$$

Here $p_{\theta|x,g}$ denotes the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{x}, g)$. This identity holds for any distribution $q_\theta(\boldsymbol{\theta})$.

Rearranging (A.1.16) gives:

$$F^{\mathrm{NM}}(q_\theta, g) = l(g) - D_{KL}(q_\theta || p_{\theta|x,g}). \tag{A.1.19}$$

Since $D_{KL}(q_\theta||p_{\theta|x,g}) \geq 0$, with equality when $q_\theta = p_{\theta|x,g}$, $F^{\mathrm{NM}}(q_\theta, g)$ is maximized over q_θ by setting $q_\theta = p_{\theta|x,g}$. Further

$$\max_{q_\theta} F^{\mathrm{NM}}(q_\theta, g) = l(g), \tag{A.1.20}$$

so

$$\arg\max_{g\in\mathcal{G}} \max_{q_\theta} F^{\mathrm{NM}}(q_\theta, g) = \arg\max_{g\in\mathcal{G}} l(g) = \hat{g}. \tag{A.1.21}$$

It remains only to show that F^{NM} has the form (A.1.11).

By (A.1.5) and (A.1.7), we have

$$\log p(\boldsymbol{x}, \boldsymbol{\theta}|g) = -\frac{1}{2}\sum_j s_j^{-2}(x_j - \theta_j)^2 + \sum_j \log g(\theta_j) + \text{const.} \tag{A.1.22}$$

Thus

$$F^{\mathrm{NM}}(q_\theta, g) = E_{q_\theta}\left[-\frac{1}{2}\sum_j (A_j\theta_j^2 - 2B_j\theta_j)\right] + \sum_j E_{q_\theta}\log g(\theta_j) - E_{q_\theta}\log q_\theta(\boldsymbol{\theta}) + \text{const.} \tag{A.1.23}$$

□

A.2 Objective function computation

Lemma 2. *Suppose $\hat{g}, q$ solves the EBNM problem with data $(\boldsymbol{x}, \boldsymbol{s})$:*

$$(\hat{g}, q) = EBNM(\boldsymbol{x}, \boldsymbol{s}), \tag{A.2.1}$$

where $q := (q_1, \ldots, q_n)$ are the estimated posterior distributions of the normal means parameters $\theta_1, \ldots, \theta_n$. Then

$$E_q(\log(\prod_j \hat{g}(\theta_j)/\prod_j q_j(\theta_j))) = l(\hat{g}; \boldsymbol{x}, \boldsymbol{s}) + \frac{1}{2}\sum_j \log(2\pi s_j^2) + (1/s_j^2)(x_j^2 + E_q(\theta_j^2) - 2x_j E_q(\theta_j) \tag{A.2.2}$$

where $l(\hat{g}; \boldsymbol{x}, \boldsymbol{s})$ is the log-likelihood for the normal means problem.

Proof. We have from (A.1.17)

$$F^{\mathrm{NM}}(q_\theta, \hat{g}) = \int q_\theta(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{x}, \boldsymbol{\theta}|\hat{g})}{q_\theta(\boldsymbol{\theta})}\, d\boldsymbol{\theta} \tag{A.2.3}$$

$$= \int q_\theta(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{x}|\boldsymbol{\theta})\hat{g}(\boldsymbol{\theta})}{q_\theta(\boldsymbol{\theta})}\, d\boldsymbol{\theta} \tag{A.2.4}$$

$$= E_q(\log(\prod_j \hat{g}(\theta_j)/\prod_j q_j(\theta_j))) - \frac{1}{2}E_q[\sum_j \log(2\pi s_j^2) + (1/s_j^2)(x_j - \theta_j)^2]. \tag{A.2.5}$$

And the result follows from noting that $F^{\mathrm{NM}}(\hat{q}, \hat{g}) = l(\hat{g})$. □

A.3 Updates of K-factors model

For FLASH model with multiple factors, we do inference by each factor. Take k_{th} factor as example, the objective function is:

$$\arg\max_{q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}} F(q_{\boldsymbol{l}_k}, q_{\boldsymbol{l}_{-k}}, q_{\boldsymbol{f}_k}, q_{\boldsymbol{f}_{-k}}, g_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_{-k}}, g_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_{-k}}, \sigma^2)$$

which can be simplified as $F^{\boldsymbol{l}_k}(q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}, \sigma)$ given all $q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}$ and all other $q_{\boldsymbol{l}_{-k}}, g_{\boldsymbol{l}_{-k}}$. Following the similar derivation above,

$$\begin{aligned} F^{\boldsymbol{l}_k}(q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}) &= E_{q_{\boldsymbol{l}_k}}\left[-\frac{1}{2}\sum_i (A_{ik} l_{ik}^2 - 2B_{ik} l_{ik})\right] \\ &+ E_{q_{\boldsymbol{l}}} p(\boldsymbol{l}_k | g_{\boldsymbol{l}_k}) - E_{q_{\boldsymbol{l}}} \log q_{\boldsymbol{l}}(\eta^{\boldsymbol{l}}, Z^{\boldsymbol{l}}) + C_1 \end{aligned} \tag{A.3.1}$$

where C_1 is a constant with respect to q_{l_k}, g_{l_k} and

$$A_{ik} = \sum_j \tau_{ij} \mathrm{E}_{q_{\boldsymbol{f}}}(f_{kj}^2) \tag{A.3.2}$$

$$B_{ik} = \sum_j \tau_{ij} \left(R_{ij}^k \mathrm{E}_{q_{\boldsymbol{f}}} f_{kj}\right) \tag{A.3.3}$$

where $R_{ij}^k := Y_{ij} - \sum_{k' \neq k} \bar{l}_{ik'} \bar{f}_{k'j}$ and $\bar{l}_{k'i} = E_{q_{\boldsymbol{l}_{-k}}} l_{ik'}; \bar{f}_{k'j} = E_{q_{\boldsymbol{f}_{-k}}} f_{k'j}$.

Based on Lemma 1, we can solve this optimization problem by the Empirical

Bayes solution to the normal means problem with:

$$x_i = \frac{\sum_j \tau_{ij} \left(R_{ij}^k \mathrm{E}_{q_{\boldsymbol{f}}} f_{kj} \right)}{\sum_j \tau_{ij} \mathrm{E}_{q_{\boldsymbol{f}}} (f_{kj}^2)} \tag{A.3.4}$$

$$s_i^2 = \frac{1}{\sum_j \tau_{ij} \mathrm{E}_{q_{\boldsymbol{f}}} (f_{kj}^2)}. \tag{A.3.5}$$

Similarly, based on Lemma 1,the updates of $q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}$ for k_{th} factor to optimize the following objective function:

$$\arg \max_{q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}} F(q_{\boldsymbol{l}_k}, q_{\boldsymbol{l}_{-k}}, q_{\boldsymbol{f}_k}, q_{\boldsymbol{f}_{-k}}, g_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_{-k}}, g_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_{-k}}, \sigma^2) \tag{A.3.6}$$

can also be solved by the Empirical Bayes solution to the normal means problem with:

$$x_j = \frac{\sum_i \tau_{ij} \left(R_{ij}^k \mathrm{E}_{q_{\boldsymbol{l}}} l_{ik} \right)}{\sum_i \tau_{ij} \mathrm{E}_{q_{\boldsymbol{l}}} (l_{ik}^2)} \tag{A.3.7}$$

$$s_j^2 = \frac{1}{\sum_i \tau_{ij} \mathrm{E}_{q_{\boldsymbol{l}}} (l_{ik}^2)}. \tag{A.3.8}$$

A.4 Algorithmic details

We implement two algorithms for the K-factor EBFA model.

Greedy Algorithm

The greedy algorithm is a forward procedure by including one possible rank one matrix $\boldsymbol{l}_k \boldsymbol{f}_k^T$ each time. The rank K model is obtained by repeatedly estimate each

pair of $\boldsymbol{l}_k, \boldsymbol{f}_k$ until the procedure stops. The stopping criteria in our VEM framework is that $g_{\boldsymbol{l}_k}, g_{\boldsymbol{f}_k}$ for any new added pair, $\boldsymbol{l}_k$ and $\boldsymbol{f}_k$, turn out to be point mass function at zero, which implies the approximated posterior distributions are equal to zero with probability one. The algorithm as follows:

Algorithm 7 Fitting K-factor EBFA model (Greedy)

for $k = 1, \cdots, K$ **do**

  $t = 0$

  initialize $q^{(0)}_{\boldsymbol{l}_k}$, $q^{(0)}_{\boldsymbol{f}_k}$, $g^{(0)}_{\boldsymbol{l}_k}$, $g^{(0)}_{\boldsymbol{f}_k}$ and $\sigma^2_{(0)}$

  $\epsilon = 1$

  while $\epsilon > 10^{-5}$ **do**

    $t = t + 1$

    $q^{(t)}_{\boldsymbol{l}_k}, g^{(t)}_{\boldsymbol{l}_k} \leftarrow \underset{q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}}{\arg\max} F(q_{\boldsymbol{l}_k}, q^{(t-1)}_{\boldsymbol{l}_{1\cdots k-1}}, q^{(t-1)}_{\boldsymbol{f}_k}, q^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, g_{\boldsymbol{l}_k}, g^{(t-1)}_{\boldsymbol{l}_{1\cdots k-1}}, g^{(t-1)}_{\boldsymbol{f}_k}, g^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, \sigma^2_{(t-1)})$

    $q^{(t)}_{\boldsymbol{f}_k}, g^{(t)}_{\boldsymbol{f}_k} \leftarrow \underset{q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}}{\arg\max} F(q^{(t)}_{\boldsymbol{l}_k}, q^{(t-1)}_{\boldsymbol{l}_{1\cdots k-1}}, q_{\boldsymbol{f}_k}, q^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, g^{(t)}_{\boldsymbol{l}_k}, g^{(t-1)}_{\boldsymbol{l}_{-k}}, g_{\boldsymbol{f}_k}, g^{(t-1)}_{\boldsymbol{f}_{1\cdots k-1}}, \sigma^2_{(t-1)})$

    $\sigma^2_{(t)} \leftarrow \underset{\sigma^2}{\arg\max} F(q^{(t)}_{\boldsymbol{l}_k}, q^{(t)}_{\boldsymbol{l}_{1\cdots k-1}}, q^{(t)}_{\boldsymbol{f}_k}, q^{(t)}_{\boldsymbol{f}_{1\cdots k-1}}, g^{(t)}_{\boldsymbol{l}_k}, g^{(t)}_{\boldsymbol{l}_{1\cdots k-1}}, g^{(t)}_{\boldsymbol{f}_k}, g^{(t)}_{\boldsymbol{f}_{1\cdots k-1}}, \sigma^2)$

    value for stopping criteria:

$$\begin{aligned} \epsilon &= F(q^{(t)}_{\boldsymbol{l}_{1,\cdots,k}}, q^{(t)}_{\boldsymbol{f}_{1,\cdots,k}}, \pi^{\boldsymbol{l}_{1,\cdots,k}(t)}, \pi^{\boldsymbol{f}_{1,\cdots,k}(t)}) \\ &- F(q^{(t-1)}_{\boldsymbol{l}_{1,\cdots,k}}, q^{(t-1)}_{\boldsymbol{f}_{1,\cdots,k}}, \pi^{\boldsymbol{l}_{1,\cdots,k}(t-1)}, \pi^{\boldsymbol{f}_{1,\cdots,k}(t-1)}) \end{aligned}$$

    if $g_{\boldsymbol{l}_k} == \delta(0)$ or $g_{\boldsymbol{f}_k} == \delta(0)$ **then**

      $\tilde{K} = k - 1$

      break

return $list(q_{\boldsymbol{l}_1}, q_{\boldsymbol{f}_1}, \cdots, q_{\boldsymbol{l}_{\tilde{K}}}, q_{\boldsymbol{f}_{\tilde{K}}}; g_{\boldsymbol{l}_1}, g_{\boldsymbol{f}_1}, \cdots, g_{\boldsymbol{l}_{\tilde{K}}}, g_{\boldsymbol{f}_{\tilde{K}}}; \sigma^2)$

Backfitting Algorithm

Backfitting algorithm (Breiman and Friedman, 1985) is a simple iterative approach to fit additive model. The optimization of current factor and loading given all others is repeated for each k, and any zero estimation of $\boldsymbol{l}_k$ or $\boldsymbol{f}_k$ (the posterior distribution is zero with probability one) would be dropped out. The algorithm is described in Algorithm 8. This approach can run for any initial value of rank K structure and refine the estimations. For example, we can take the results from greedy algorithm as initial value of the rank K structure.

The details of the algorithm as follows:

Algorithm 8 Fitting K-factor EBFA model (Backfitting)

1: $t = 0$

2: initialize $q^0_{\boldsymbol{l}_1}, \cdots, q^0_{\boldsymbol{l}_K}$, $q^0_{\boldsymbol{f}_1}, \cdots, q^0_{\boldsymbol{f}_K}$, $g^0_{\boldsymbol{l}_1}, \cdots, g^0_{\boldsymbol{l}_K}$, $g^0_{\boldsymbol{f}_1} \cdots, g^0_{\boldsymbol{f}_K}$ and $\sigma^2_{(0)}$

3: $\epsilon = 1$

4: **while** $\epsilon > 10^{-5}$ **do**

5: $t = t + 1$

6: **for** $k = 1, \cdots, K$ **do**

7: $q^{(t)}_{\boldsymbol{l}_k}, g^{(t)}_{\boldsymbol{l}_k} \leftarrow \underset{q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}}{\arg\max} F(q_{\boldsymbol{l}_k}, q^{(t-1)}_{\boldsymbol{l}_{-k}}, q^{(t-1)}_{\boldsymbol{f}_k}, q^{(t-1)}_{\boldsymbol{f}_{-k}}, g_{\boldsymbol{l}_k}, g^{(t-1)}_{\boldsymbol{l}_{-k}}, g^{(t-1)}_{\boldsymbol{f}_k}, g^{(t-1)}_{\boldsymbol{f}_{-k}}, \sigma^2_{(t-1)})$

8: $q^{(t)}_{\boldsymbol{f}_k}, g^{(t)}_{\boldsymbol{f}_k} \leftarrow \underset{q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}}{\arg\max} F(q^{(t)}_{\boldsymbol{l}_k}, q^{(t-1)}_{\boldsymbol{l}_{-k}}, q_{\boldsymbol{f}_k}, q^{(t-1)}_{\boldsymbol{f}_{-k}}, g^{(t)}_{\boldsymbol{l}_k}, g^{(t-1)}_{\boldsymbol{l}_{-k}}, g_{\boldsymbol{f}_k}, g^{(t-1)}_{\boldsymbol{f}_{-k}}, \sigma^2_{(t-1)})$

9: $\sigma^2_{(t)} \leftarrow \underset{\sigma^2}{\arg\max} F(q^{(t)}_{\boldsymbol{l}_k}, q^{(t)}_{\boldsymbol{l}_{-k}}, q^{(t)}_{\boldsymbol{f}_k}, q^{(t)}_{\boldsymbol{f}_{-k}}, g^{(t)}_{\boldsymbol{l}_k}, g^{(t)}_{\boldsymbol{l}_{-k}}, g^{(t)}_{\boldsymbol{f}_k}, g^{(t)}_{\boldsymbol{f}_{-k}}, \sigma^2)$

10: value for stopping criteria:

$$\begin{aligned} \epsilon &= F(q^{(t)}_{\boldsymbol{l}_{1,\cdots,K}}, q^{(t)}_{\boldsymbol{f}_{1,\cdots,K}}, \pi^{\boldsymbol{l}_{1,\cdots,K}(t)}, \pi^{\boldsymbol{f}_{1,\cdots,K}(t)}) \\ &- F(q^{(t-1)}_{\boldsymbol{l}_{1,\cdots,K}}, q^{(t-1)}_{\boldsymbol{f}_{1,\cdots,K}}, \pi^{\boldsymbol{l}_{1,\cdots,K}(t-1)}, \pi^{\boldsymbol{f}_{1,\cdots,K}(t-1)}) \end{aligned}$$

return $list(q_{\boldsymbol{l}_1}, q_{\boldsymbol{f}_1}, \cdots, q_{\boldsymbol{l}_K}, q_{\boldsymbol{f}_K}; g_{\boldsymbol{l}_1}, g_{\boldsymbol{f}_1}, \cdots, g_{\boldsymbol{l}_K}, g_{\boldsymbol{f}_K}; \sigma^2)$

Following the same derivation as rank one model, the step 8 and step 9 in Algorithm 8 to update the $q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}$ and $q_{\boldsymbol{f}_k}, g_{\boldsymbol{f}_k}$ can be solved by EB approach for normal means problem.

A.5 Estimates of σ

We now discuss this problem for various choices of Σ.

Constant Variance $\sigma_{ij} = \sigma$

$$\hat{\sigma}^2 = \frac{1}{NP} \sum_{ij} \bar{R}_{ij}^2. \tag{A.5.1}$$

Column-wise Heteroskedastic Error $\sigma_{ij} = \sigma_j$

$$\hat{\sigma}_j^2 = \frac{1}{N} \sum_{i} \bar{R}_{ij}^2. \tag{A.5.2}$$

Kronecker Product Structure ($\sigma_{ij} = 1/\tau_i^l \tau_j^f$)

In this case there is a non-identifiability since we can multiply every τ_i^l by some constant c and divide every τ_j^f by c the value of σ_{ij} remains the same. This can be deal with by imposing an arbitrary identifiability constraint (e.g. $\sum_j \tau_j^f = 1$). (Alternatively we could use a proper prior on both τ^l and τ^f to avoid the identifiability; the results assume independent Gamma priors $G(\alpha, \beta)$ on τ_i^l and τ_j^f.)

As far as we know there is no closed-form solution to the joint optimization over (τ^l, τ^f). However, there are closed form solutions for optimizing over τ^l keeping τ^f fixed, and vice versa.

$$\hat{\tau}_i^l = \frac{P + 2\alpha}{\sum_j \bar{R}_{ij}^2 \tau_j^f + 2\beta} \tag{A.5.3}$$

$$\hat{\tau}_j^f = \frac{P + 2\alpha}{\sum_i \bar{R}_{ij}^2 \tau_i^l + 2\beta}. \tag{A.5.4}$$

We optimize F by iterating between these updates.

Noisy observations with known error variance

Now assume we have observations Y_{ij} that are "noisy" observations of underlying "true" values that are assumed to come from the EBFA model. Assume further that the noise has known standard deviation (α_{ij}^2). That is

$$Y_{ij} = Y_{ij}^{\text{true}} + N(0, \alpha_{ij}^2) \tag{A.5.5}$$

where Y^{true} follows the EBFA and α_{ij} is known. Under this assumption Y_{ij} also comes from the EBFA, with variance $\sigma_{ij}^2 = \tilde{\sigma}_{ij}^2 + \alpha_{ij}^2$ where $\tilde{\sigma}$ denotes the variance terms of the EBFA for Y^{true}.

Since α is known, optimizing over σ is equivalent to optimizing over $\tilde{\sigma}$. One might wish to make various assumptions about the form for $\tilde{\sigma}$. In general there is no closed form solution to optimizing over $\tilde{\sigma}$, but we have implemented numerical solutions to this optimization (using the R function 'optim()'), for the cases $\tilde{\sigma}_{ij} = \tilde{\sigma}$ and $\tilde{\sigma}_{ij} = \tilde{\sigma}_j$. Thus, for example, we implement numerical solution to:

$$\hat{\tilde{\sigma}}_j = \arg\max_{\tilde{\sigma}_j} \left\{ -\sum_i \log(\alpha_{ij}^2 + \tilde{\sigma}_j^2) - \sum_i \frac{\bar{R}_{ij}^2}{\alpha_{ij}^2 + \tilde{\sigma}_j^2} \right\}. \tag{A.5.6}$$

A.6 Inference with penalty term

We can also add penalty terms to the log-likelihood function (2.3.6) to make the estimation less "conservative". We denote $h_{\boldsymbol{l}}(g_{\boldsymbol{l}})$ and $h_{\boldsymbol{f}}(g_{\boldsymbol{f}})$ as the penalty terms

on $g_{\boldsymbol{l}}$ and $g_{\boldsymbol{f}}$. The regularized likelihood would be:

$$\begin{aligned} l(g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \sigma) &:= \log[p(Y|g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \sigma^2)] + h_{\boldsymbol{l}}(g_{\boldsymbol{l}}) + h_{\boldsymbol{f}}(g_{\boldsymbol{f}}) \\ &= F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \sigma^2) + h_{\boldsymbol{l}}(g_{\boldsymbol{l}}) + h_{\boldsymbol{f}}(g_{\boldsymbol{f}}) + D_{KL}(q||p) \end{aligned} \tag{A.6.1}$$

where $F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \sigma^2)$ and $D_{KL}(q||p)$ are defined in (2.3.8) and (2.3.9). Thus, the VEM algorithm with penalty terms becomes:

$$\max F(q, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \sigma^2) + h_{\boldsymbol{l}}(g_{\boldsymbol{l}}) + h_{\boldsymbol{f}}(g_{\boldsymbol{f}}). \tag{A.6.2}$$

Following the same argument in Appendix A.1, we need to solve step 7 and 8 in Algorithm 1 with penalty term in the objective function $F(\cdot)$. We start with step 7 and the penalized objective function is as follows:

$$\begin{aligned} &F(q_{\boldsymbol{l}}, q_{\boldsymbol{f}}, g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \sigma^2) + h_{\boldsymbol{l}}(g_{\boldsymbol{l}}) + h_{\boldsymbol{f}}(g_{\boldsymbol{f}}) \\ &= \int q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f}) \log \frac{p(Y, \boldsymbol{l}, \boldsymbol{f}; g_{\boldsymbol{l}}, g_{\boldsymbol{f}}, \sigma^2)}{q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{f}}(\boldsymbol{f})} + h_{\boldsymbol{l}}(g_{\boldsymbol{l}}) + h_{\boldsymbol{f}}(g_{\boldsymbol{f}}) \\ &= E_{q_{\boldsymbol{l}}} \left[-\frac{1}{2} \sum_i (A_i l_i^2 - 2 B_i l_i) \right] + \sum_i E_{q_{\boldsymbol{l}}} \log g_{\boldsymbol{l}}(l_i) - E_{q_{\boldsymbol{l}}} \log q_{\boldsymbol{l}}(\boldsymbol{l}) + h_{\boldsymbol{l}}(g_{\boldsymbol{l}}) + C_0' \end{aligned} \tag{A.6.3}$$

where C_0' is a constant with respect to $q_{\boldsymbol{l}}$ and $g_{\boldsymbol{l}}$, and

$$A_i = \frac{1}{\sigma^2} \sum_j E_{q_{\boldsymbol{f}}} f_j^2 \tag{A.6.4}$$

$$B_i = \frac{1}{\sigma^2} \sum_j (Y_{ij} E_{q_{\boldsymbol{f}}} f_j). \tag{A.6.5}$$

We now show that maximizing this over $q_{\boldsymbol{l}}$ and $g_{\boldsymbol{l}}$ comes down to solving the Empirical Bayes version of the normal means problem with penalty term.

Penalized normal means problem

With the same model of normal means problem (A.1.5) in Appendix A.1, the Empirical Bayes solution to the penalized normal means problem estimates g by maximum penalized log-likelihood:

$$\hat{g} = \arg\max_{g \in \mathcal{G}} l(g) + h_g(g), \tag{A.6.6}$$

where $l(g) = \log p(\boldsymbol{x}|g)$ and $h_g(g)$ is penalty term. And then finds the posterior distribution of $\boldsymbol{\theta}$ given $\hat{g}$:

$$p(\boldsymbol{\theta}|\boldsymbol{x}, \hat{g}) = \prod_j p(\theta_j|\boldsymbol{x}, \hat{g}) \propto \prod_j \hat{g}(\theta_j) p(x_j|\theta_j, s_j). \tag{A.6.7}$$

Lemma 3. *The Empirical Bayes solution to the penalized normal means problem also solves:*

$$\max_{q_\theta, g \in \mathcal{G}} F^{NM}(q_\theta, g) + h_g(g) \tag{A.6.8}$$

where $h_g(g)$ is penalty term and $F^{NM}(q_\theta, g)$ is defined in (A.1.11)*:*

$$F^{NM}(q_\theta, g) = E_{q_\theta}\left[-\frac{1}{2}\sum_j (A_j\theta_j^2 - 2B_j\theta_j)\right] + \sum_j \log g(\theta_j) - E_{q_\theta} \log q_\theta(\boldsymbol{\theta}) + const$$

Equivalently, (A.6.8) *is solved by $g = \hat{g}$ in* (A.6.6) *and $q_\theta = p(\boldsymbol{\theta}|\boldsymbol{x}, \hat{g})$ in* (A.6.7)*, with $x_j = B_j/A_j$ and $s_j^2 = 1/A_j$.*

Proof. Rearranging (A.1.16) gives:

$$F^{\mathrm{NM}}(q_\theta, g) + h_g(g) = l(g) + h_g(g) - D_{KL}(q_\theta || p_{\theta|x,g}), \tag{A.6.9}$$

where $F^{\mathrm{NM}}(q_\theta, g)$ and $D_{KL}(q_\theta || p_{\theta|x,g})$ are given in (A.1.17)-(A.1.18).

Since $D_{KL}(q_\theta || p_{\theta|x,g}) \geq 0$, with equality when $q_\theta = p_{\theta|x,g}$, $F^{\mathrm{NM}}(q_\theta, g)$ is maximized over q_θ by setting $q_\theta = p_{\theta|x,g}$. Similarly with (A.1.20), we can get:

$$\max_{q_\theta} F^{\mathrm{NM}}(q_\theta, g) + h_g(g) = l(g) + h_g(g), \tag{A.6.10}$$

so

$$\arg\max_{g \in \mathcal{G}} \max_{q_\theta} F^{\mathrm{NM}}(q_\theta, g) + h_g(g) = \arg\max_{g \in \mathcal{G}} l(g) + h_g(g) = \hat{g}. \tag{A.6.11}$$

It remains only to show that F^{NM} has the form (A.1.11).

Based on the (A.1.23) in proof in Lemma 1, we can get that:

$$F^{\mathrm{NM}}(q_\theta, g) = E_{q_\theta}\left[-\frac{1}{2}\sum_j (A_j\theta_j^2 - 2B_j\theta_j)\right] + \sum_j E_{q_\theta} \log g(\theta_j) - E_{q_\theta} \log q_\theta(\boldsymbol{\theta}) + \text{const.} \tag{A.6.12}$$

□

Based on Lemma 3, the step 7 in Algorithm 1 with penalty term can be solved by EB approach on penalized normal means problem with (2.3.14)-(2.3.15). Similarly the solution of EB approach for penalized normal means problem can also solve step 8 in Algorithm 1 with penalty term.

In this subsection, we mainly focus on the rank one case, and it is easy to extend the whole procedure to the case of multiple factors with penalty terms.

A.7 Fixed Factor model

In this case, we try to fit factor model with factor observed. The details of FLASH with fixed factor is described as follows:

Algorithm 9 FLASH fixed factor

procedure VARIATIONAL EM ALGORITHM

  $t = 0$

  initialize $q_{\boldsymbol{l}}^{(0)}$, $g_{\boldsymbol{l}}^{(0)}$ and calculate $\sigma^{(0)2}$

  $\epsilon = 1$

  while $\epsilon > 1e-6$ **do**

    $t = t + 1$

    $q_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{l}}^{(t)} \leftarrow \arg\max_{q_{\boldsymbol{l}}, g_{\boldsymbol{l}}} F(q_{\boldsymbol{l}}^{(t-1)}, g_{\boldsymbol{l}}^{(t-1)}, \sigma^{(t-1)2})$

    $\sigma^{(t)2} \leftarrow \arg\max_{(}\, \sigma) F(q_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{l}}^{(t)}, \sigma^2)$

    $\epsilon = F(q_{\boldsymbol{l}}^{(t)}, g_{\boldsymbol{l}}^{(t)}, \sigma^{(t)2}) - F(q_{\boldsymbol{l}}^{(t-1)}, g_{\boldsymbol{l}}^{l(t-1)}, \sigma^{(t-1)2})$

  return $list(q_{\boldsymbol{l}}, g_{\boldsymbol{l}})$

APPENDIX B

SUPPLEMENTARY FOR LOGISTIC FLASH

B.1 Variational Inference for rank K Binomial model

In the variational inference, our goal is to maximize the ELBO as follows:

$$
\begin{aligned}
&F(q_{\boldsymbol{l}_{1,\cdots,K}}, g_{\boldsymbol{l}_{1,\cdots,K}}, q_{\boldsymbol{f}_{1,\cdots,K}}, g_{\boldsymbol{f}_{1,\cdots,K}}, q_\omega) \qquad \text{(B.1.1)}\\
&\equiv \iiint \prod_k q_{\boldsymbol{f}_k}(\boldsymbol{f}) q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_\omega(\omega) \log \frac{p(Y,\omega|L,F)\prod_k p(\boldsymbol{l}_k|g_{\boldsymbol{l}_k}) p(\boldsymbol{f}_k|g_{f_k})}{\prod_k q_{\boldsymbol{f}_k}(\boldsymbol{f}) q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) q_\omega(\omega)} d\boldsymbol{f} d\boldsymbol{l} d\omega
\end{aligned}
$$

where ω is $N \times P$ matrix. The variational approximation for the posterior distribution of elements of ω is:

$$
\begin{aligned}
q_{\omega_{ij}}(\omega_{ij}) &\propto \exp\left\{ E_{q_{\boldsymbol{l}}, q_{\boldsymbol{f}}} log(p(Y_{ij}, \omega_{ij}|L,F) \prod_k p(\boldsymbol{l}_k|g_{\boldsymbol{l}_k}) p(\boldsymbol{f}_k|g_{f_k})) \right\} \qquad \text{(B.1.2)}\\
&= \exp\left\{ -\frac{\sum_k E_q l_{ik}^2 E_q f_{kj}^2 \omega_{ij}}{2} + \log PG(\omega_{ij}|n_{ij}, 0) \right\}\\
&\propto PG(\omega_{ij}|n_{ij}, \sqrt{\xi_{ij}})
\end{aligned}
$$

where $\xi_{ij} = E_q(\sum_k l_{ik} E_q f_{kj})^2$ and $q(\omega) = \prod_{ij} q_{\omega_{ij}}(\omega_{ij})$. We can see that the variational approximation for each element of ω follows Polya-Gamma distribution and the first moment is easy to get:

$$
E_q \omega_{ij} = \frac{n_{ij}}{2\xi_{ij}} \tanh(\frac{\xi_{ij}}{2}). \qquad \text{(B.1.3)}
$$

Using the coordinate ascent procedure, we repeatedly optimize over each pair of

$q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}$ and $q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}$ given others fixed. We start from the inference of $q_{\boldsymbol{l}_k}, g_{\boldsymbol{l}_k}$. The ELBO can be simplified given the current updates of $q_{\boldsymbol{l}_{-k}}, g_{\boldsymbol{l}_{-k}}, q_{\boldsymbol{f}_{1,\cdots,K}}, g_{\boldsymbol{l}_{1,\cdots,K}}$:

$$
\begin{aligned}
&F(q_{\boldsymbol{l}_{1,\cdots,K}}, g_{\boldsymbol{l}_{1,\cdots,K}}, q_{\boldsymbol{f}_{1,\cdots,K}}, g_{\boldsymbol{f}_{1,\cdots,K}}, q_\omega) \qquad \text{(B.1.4)}\\
&= E_q \log(p(Y,\omega|L,F) + E_q p(\boldsymbol{l}_k|g_{\boldsymbol{l}_k}) - E_q q_{\boldsymbol{l}_k}(\boldsymbol{l}_k) + C_l''\\
&= \sum_{ij}\left\{-\frac{E_{q_{-\boldsymbol{l}_k}}(\sum_k l_{ik}f_{kj})^2 E_q\omega_{ij}}{2} + \tilde{Y}_{ij}l_{ik}E_q f_{kj}\right\} - E_q q_{\boldsymbol{l}}(\boldsymbol{l}_k) + E_q p(\boldsymbol{l}_k|g_{\boldsymbol{l}}) + C_l'''
\end{aligned}
$$

where C_l'' and C_l''' are constant with respect to $\boldsymbol{l}_k$, and

$$E_q\omega_{ij} = \frac{n_{ij}}{2\xi_{ij}}\tanh(\frac{\xi_{ij}}{2}).$$

The first term of objective function in (B.1.4) is

$$E_q \log(p(Y,\omega|L,F) = -\frac{(l_{ik}^2 E_q f_{kj}^2 + 2l_{ik}E_q f_{kj}(\sum_{k'\neq k} E_q l_{ik'}E_q f_{k'j}))E_q\omega_{ij}}{2} + \tilde{Y}_{ij}l_{ik}E_q f_{kj}$$

which is a quadratic form of $\boldsymbol{l}_k$, so we can apply EB approach for normal means problem with parameters:

$$x_i = \frac{\sum_j(\tilde{Y}_{ij} - E_q\omega_{ij}\sum_{t\neq k} El_{it}Ef_{tj})Ef_{kj}}{\sum_j(E_q\omega_{ij}Ef_{kj}^2)} \qquad \text{(B.1.5)}$$

$$s_i^2 = \frac{1}{\sum_j(E_q\omega_{ij}Ef_{kj}^2)}. \qquad \text{(B.1.6)}$$

The updates in (B.1.5)-(B.1.6) are also the same as the updates in (3.2.25)-(3.2.26) when fixing $n_{ij} = 1$ and transforming the Y_{ij} into same scale in both proce-

dures.

Similarly, the updates for $\boldsymbol{f}_k$ can be obtained from EB approach for normal means problem with parameters:

$$x_j = \frac{\sum_i (\tilde{Y}_{ij} - E_q\omega_{ij} \sum_{t\neq k} El_{it} Ef_{tj}) El_{ik}}{\sum_i (E_q\omega_{ij} El_{ik}^2)} \tag{B.1.7}$$

$$s_j^2 = \frac{1}{\sum_i (E_q\omega_{ij} El_{ik}^2)}. \tag{B.1.8}$$

Since the updates in this procedure are the same as (3.2.25)-(3.2.28) after some transformations, so the variational algorithm should also be the same as Algorithm 4. We just need to substitute the updates with more general version (B.1.5)-(B.1.8). We can also use greedy algorithm by iteratively including one factor at a time, where the residual matrix used in initialization of factor and loading for each k is $\tilde{Y}_{ij} - E_q\omega_{ij} \sum_{t\neq k} El_{it} Ef_{tj}$.

B.2 Inference for Possion model

In Possion data case, we can reparameterize the model (3.5.1) as follows:

$$\begin{aligned} Y_{ij}|n_{ij}, p_{ij} &= Bin(n_{ij}, p_{ij}) && \text{(B.2.1)} \\ \log \frac{p_{ij}}{1-p_{ij}} &= l_i f_j && \text{(B.2.2)} \\ n_{ij} &\sim Poisson(\lambda_{ij}). && \text{(B.2.3)} \end{aligned}$$

In this reparameterization, we consider the data following compound Possion distribution. This model is equivalent to the Poisson model (3.5.1) and we will introduce

inference based on this reparameterized model.

For the inference, we start from the simplest case, rank one model, and provide a brief procedure in three steps:

1. $p_{ij}|Y_{ij}, n_{ij}$:

   we can apply logistic flash on p_{ij} given n_{ij} because the problem is a logistic flash model given n_{ij}:

$$\begin{aligned} Y_{ij}|n_{ij}, p_{ij} &= Bin(n_{ij}, p_{ij}) && \text{(B.2.4)} \\ \log \frac{p_{ij}}{1-p_{ij}} &= l_i f_j. && \text{(B.2.5)} \end{aligned}$$

2. $n_{ij}|p_{ij}, Y_{ij}, \lambda_{ij}$:

   The marginal disctribution of n_{ij} given p_{ij} is

$$n_{ij} \sim Y_{ij} + poisson(\lambda_{ij}(1-p_{ij})) \qquad \text{(B.2.6)}$$

   which can be calculated as follows

$$\begin{aligned} p(n_{ij}|p_{ij}, \lambda_{ij}, Y_{ij}) &\propto p_{ij}^{Y_{ij}}(1-p_{ij})^{n_{ij}-Y_{ij}} \frac{n_{ij}!}{Y_{ij}!(n_{ij}-Y_{ij})!} \frac{\lambda_{ij}^{n_{ij}} e^{-\lambda_{ij}}}{n_{ij}!} \qquad \text{(B.2.7)} \\ &\propto \frac{(\lambda_{ij}(1-p_{ij}))^{(n_{ij}-Y_{ij})} e^{-(\lambda_{ij}(1-p_{ij}))}}{(n_{ij}-Y_{ij})!}. \end{aligned}$$

3. $\lambda_{ij}|p_{ij}, Y_{ij}$:

This is the model for *ash* method on poisson likelihood:

$$Y_{ij} \sim poisson(\lambda_{ij} p_{ij}) \tag{B.2.8}$$

$$\lambda_{ij} \sim g_\lambda(\cdot). \tag{B.2.9}$$

In this case, we can apply *ash* to get the posterior distribution.

The likelihood of compound Poisson model is:

$$\begin{aligned} p(Y) &= \iiint p(Y|\lambda, \boldsymbol{l}, \boldsymbol{f}) p(\boldsymbol{l}|g_{\boldsymbol{l}}) p(\boldsymbol{f}|g_{\boldsymbol{f}}) p(\lambda|g_\lambda) \\ &= \iiint \prod_{ij} p(Y_{ij}|n_{ij}, p_{ij}) p(n_{ij}|\lambda_{ij}) p(l_i|g_{\boldsymbol{l}}) p(f_j|g_{\boldsymbol{f}}) p(\lambda_{ij}|g_\lambda). \end{aligned} \tag{B.2.10}$$

We can consider the above procedure as Gibbs sampling procedure, in which the samples come from conditional distributions for each variable. We modify the conditional distributions of $\boldsymbol{l}, \boldsymbol{f}|Y_i j, n_i j, \lambda_{ij}$ by using variational approximation since the explicit form of posterior distribution is hard to calculate, so it's not exactly Gibbs sampling scheme.

B.3 Data analysis

We applied Logistic Flash to some toy examples, such as chiq-seq data, single cell data and others, and we take the MNIST data as example.

$$https://nkweiwang.github.io/logisticFlash_workflow/index.html$$

B.3.1 MNIST data set

We apply Logistic Flash to binarized digit pictures and use T-SNE (van der Maaten and Hinton, 2008) to get two-dimension visualization. MNIST data (Mixed National Institute of Standards and Technology database (LeCun et al., 1998)) is a large database of handwritten digits.

$$http://yann.lecun.com/exdb/mnist/$$

In this section, we take the binarized handwritten digits as toy example (by taking the none zero element as one). We can see that the binarized digit remains the most information out of the original picture (see Figure B.1 for example).

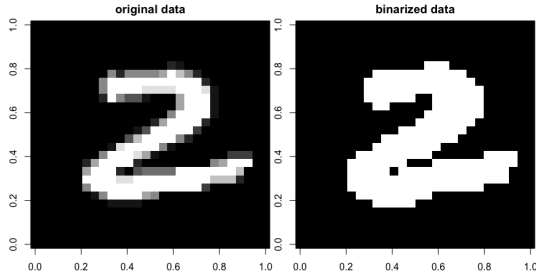

Figure B.1: The digit in binarized data can still be easily recognized.

We apply Logistic Flash on the binarized digit pictures. In this example, we select 1000 digits (100 for each digit, $0, \cdots, 9$). We use TSNE (van der Maaten and Hinton, 2008) to visualize the result from greedy algorithm with setting the maximum of rank equal to 9. To see the clustering of the digits by TSNE, we use the estimated low rank matrix excluding first factor (common factor) and visualize

it in 2D plot in Figure B.2. We can see that the data points belong to the same digit group together. So we can still capture the clustering structure using Logistic Flash after binarizing the data. The 2D visualization of SVD result with maximum rank as 9 is provided in Figure B.3. In the SVD result, we exclude the first singular vector as well.

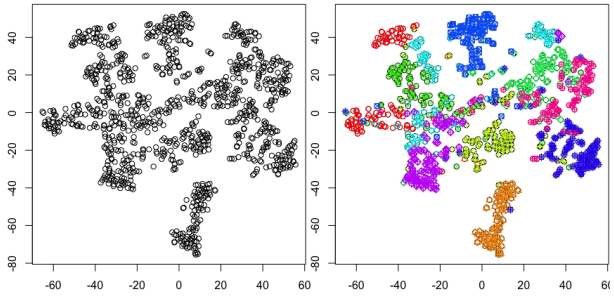

Figure B.2: TSNE result on the estimated low rank structure from Logistic Flash. The right one is colored by different digits.

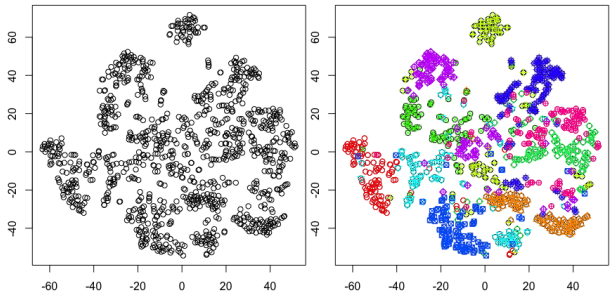

Figure B.3: TSNE result on the estimated low rank structure from SVD. The right one is colored by different digits

APPENDIX C

SUPPLEMENTARY FOR DSC OMEGA

C.1 Examples for graph patterns

There are many other types of graph provided in (Zhao et al., 2012). The "huge" package generates the data matrix corresponding to the graph. Here we show some other types of graph patterns used in our comparison. The plots are generated from "huge" R-package.

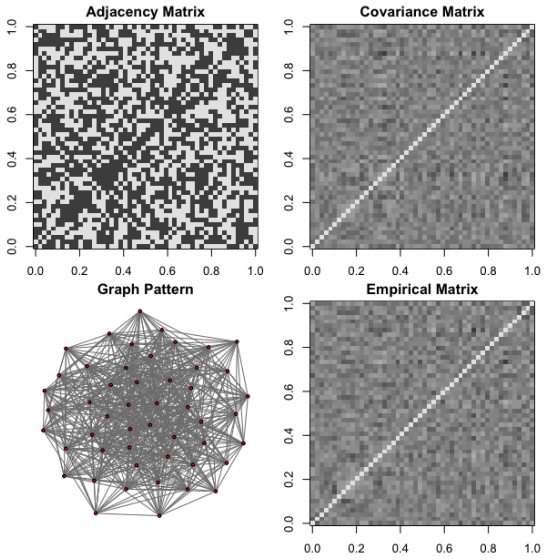

Figure C.1: Dense Edors Renyi graph with random edges.

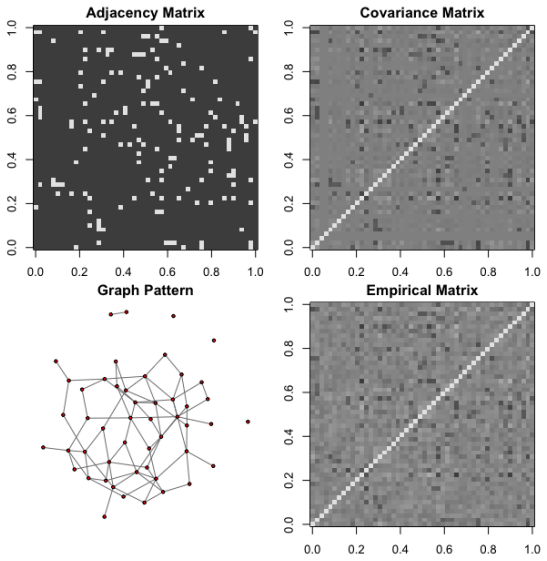

Figure C.2: Sparse Edors Renyi graph with random edges.

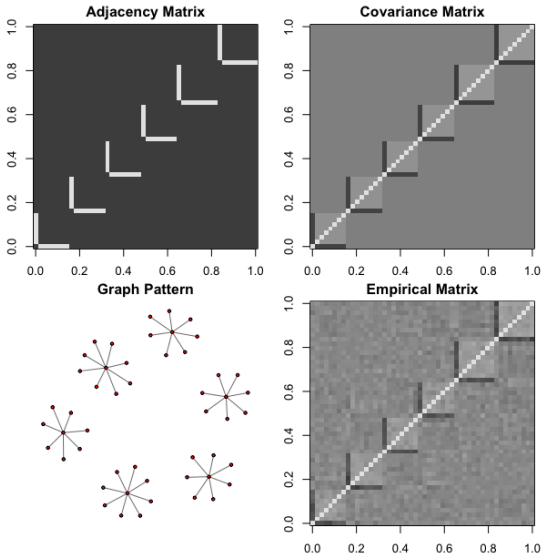

Figure C.3: Hub graph with $K = 6$ hubs.

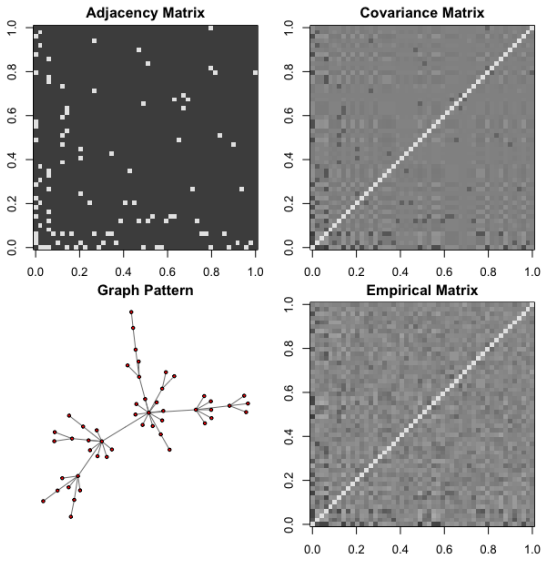

Figure C.4: Scale-free graph with P edges in the graph.

APPENDIX D

SUPPLEMENTARY FOR MVASH

D.1 Parameters update

we consider the regression model as follows:

$$\begin{aligned} y &= \mathbf{X}\beta + \varepsilon \\ &= X_1\beta_1 + \cdots + X_P\beta_P + \varepsilon \end{aligned} \tag{D.1.1}$$

in which $\varepsilon \sim N(0, \sigma_e^2)$.

D.1.1 Variational Inference

For the posterior of the β, our goal is to find an independent approximation to minimize the K-L distance between the posterior density and the independent approximation,

$$KL_{q||p}(\theta) = \iint q(\beta, \gamma; \theta) \log \left\{ \frac{q(\beta, \gamma; \theta)}{p(y|\mathbf{X}, \beta, \theta)p(\beta|\theta)} \right\} d\beta d\gamma + C. \tag{D.1.2}$$

where C is constant with respect to q.

The independent approximation is defined as follows:

$$q(\beta, \gamma; \theta) = \prod_{j=1}^{P} q_j(\beta_j, \gamma_j; \theta) \tag{D.1.3}$$

where γ_j takes values in $1, ..., K$ as an indicator for β_j falling in k^{th} class. Here we assume that the approximation of the posterior density can be written as a mixture of normal density for each component β_j, $q(\beta_j, \gamma_j; \theta)$, where $\theta \equiv \{\sigma_e^2, \sigma_2^2, \ldots, \sigma_K^2, \pi_1, \ldots, \pi_K\}$

$$q(\beta_j, \gamma_j; \theta) = [\alpha_{j1}\delta_0(\beta_j)]^{I_{(\gamma_j=1)}} \prod_{k=2}^{K} [\alpha_{jk} N(\beta_j; \mu_{jk}, s_{jk})]^{I_{(\gamma_j=k)}}. \tag{D.1.4}$$

We can see that it is a mixture of point mass at zero and different normal densities with probability of α_{jk} for every β_j, which can be simplified as follows:

$$\beta_j \sim \alpha_{j1}\delta_0 + \sum_{k=2}^{K} \alpha_{ji} N(\mu_{jk}, s_{jk})). \tag{D.1.5}$$

Recalling the prior of β we propose in this article:

$$p(\beta, \gamma|\theta) = \prod_{j=1}^{P} p(\beta_j, \gamma_j|\theta) \tag{D.1.6}$$

where

$$p(\beta_j, \gamma_j|\theta) = [\pi_1\delta_0(\beta_j)] \prod_{k=2}^{K} \pi_i N(\beta_j; 0, \sigma_k^2\sigma_e^2). \tag{D.1.7}$$

The likelihood term can be written as:

$$P(y|\mathbf{X}, \beta) = (\frac{1}{\sqrt{2\pi\sigma_e^2}})^n \exp\left\{-\frac{1}{2\sigma_e^2}(y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta)\right\}. \tag{D.1.8}$$

To get the best fully-factorized approximation for posterior, we can specify the parameters in the fully-factorized form by minimizing the K-L distance between posterior and the fully-factorized approximation. It is a good approximation when the posterior is concentrated at single location which corresponds to our unimodal assumption. By the definition of K-L distance between the posterior and approximation for the posterior, we can obtain that minimizing the K-L distance is equivalent to maximizing the lower bound of the log likelihood,

$$\log p(y \mid \mathbf{X}, \theta) \geq F(\theta) \equiv \iint q(\beta, \gamma; \theta) \log \left\{ \frac{p(y \mid \mathbf{X}, \beta, \sigma^2)\, p(\beta|\theta)}{q(\beta, \gamma; \theta)} \right\} d\beta d\gamma. \quad \text{(D.1.9)}$$

To minimize the K-L distance, we can further write the K-L distance into three terms as following:

$$\begin{aligned} KL_{q||p}(\theta) &= -\iint q(\beta, \gamma; \theta) log\{P(y|\mathbf{X}, \beta, \gamma)\} d\beta d\gamma & \text{(D.1.10)} \\ &\quad -\iint q(\beta, \gamma; \theta) log\{p(\beta, \gamma|\theta)\} d\beta d\gamma & \text{(D.1.11)} \\ &\quad +\iint q(\beta, \gamma; \theta) log\{q(\beta, \gamma; \theta)\} d\beta d\gamma & \text{(D.1.12)} \\ &\quad +C. \end{aligned}$$

Here we would like to show more details on the calculation for each term. The

first one can be decomposed into several terms:

$$\begin{aligned}
(\text{D.1.10}) \quad &= \frac{n}{2}log2\pi\sigma_e^2 + \frac{1}{2\sigma_e^2}\iint q(\beta,\gamma,\theta)\beta^T\mathbf{X}^T\mathbf{X}\beta d\beta d\gamma \\
&\quad -\frac{1}{\sigma_e^2}\iint q(\beta,\gamma,\theta)\beta^T\mathbf{X}^T y d\beta d\gamma + \frac{1}{2\sigma_e^2}y^T y \\
&= \frac{n}{2}log2\pi\sigma_e^2 + \frac{1}{2\sigma_e^2}y^T y - \frac{1}{\sigma_e^2}\sum_{j=1}^{P}\phi_j(\sum_{k=2}^{K}\alpha_{jk}\mu_{jk}) \\
&\quad +\frac{1}{2\sigma_e^2}\iint q(\beta,\gamma,\theta)\beta^T\mathbf{X}^T\mathbf{X}\beta d\beta d\gamma
\end{aligned}$$

where $\phi = \{\phi_1, ..., \phi_P\}^T = \mathbf{X}^T y$ and we let $r_j = \sum_{k=2}^{K}\alpha_{jk}\mu_{jk}$. Now the most difficult term to tackle with is the last one, so we focus on this following terms:

$$\iint q(\beta,\gamma,\theta)\beta^T\mathbf{X}^T\mathbf{X}\beta d\beta d\gamma$$

it can be written as:

$$\begin{aligned}
&\iint q(\beta,\gamma,\theta)\beta^T\mathbf{X}^T\mathbf{X}\beta d\beta d\gamma \qquad (\text{D.1.13}) \\
= &\iint q(\beta,\gamma,\theta)\sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{k=1}^{P}x_{ij}x_{ik}\beta_j\beta_k d\beta d\gamma \\
= &\sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{t=1}^{P}x_{ij}x_{it}\iint q(\beta,\gamma,\theta)\beta_j\beta_t d\beta d\gamma \\
= &\sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{t=1}^{P}x_{ij}x_{it}E\beta_j E\beta_t - \sum_{i=1}^{P}\sum_{t=1}^{P}x_{it}^2(E\beta_t)^2 + \sum_{i=1}^{P}\sum_{t=1}^{P}x_{it}^2 E\beta_t^2 \\
= &\ ||Xr||^2 - \sum_{t=1}^{P}d_t r_t^2 + \sum_{t=1}^{P}\{d_t\sum_{k=2}^{K}\alpha_{tk}(\mu_{tk}^2 + s_{tk})\}
\end{aligned}$$

where $r_j = E\beta_j$, $d_t = \sum_{i=1}^{P} x_{it}^2$ and $\phi = \{\phi_1, ..., \phi_P\}^T = \mathbf{X}^T y$.

By integrating out all the parameters, we can get that

$$\begin{aligned} \text{(D.1.10)} \quad &= \frac{n}{2\sigma_e^2} log 2\pi\sigma_e^2 + \frac{1}{2\sigma_e^2} y^T y - \frac{1}{\sigma_e^2} \sum_{j=1}^{P} \phi_j r_j \\ &\quad + \frac{1}{2\sigma_e^2} \left\{ ||Xr||^2 - \sum_{t=1}^{P} d_t r_t^2 + \sum_{t=1}^{P} [d_t \sum_{k=2}^{K} \alpha_{tk} (\mu_{tk}^2 + s_{tk})] \right\}. \end{aligned}$$

For the second term,

$$\begin{aligned} \text{(D.1.11)} \quad &= -\iint q(\beta, \gamma; \theta) \log\{p(\beta, \gamma|\pi)\} d\beta d\gamma \\ &= -\sum_{j=1}^{P} \iint q(\beta, \gamma; \theta) \log\{p(\beta_j, \gamma_j|\pi)\} d\beta d\gamma \\ &= -\sum_{j=1}^{P} \iint q_j(\beta_j, \gamma_j; \theta) \log\{p(\beta_j, \gamma_j|\pi)\} d\beta_j d\gamma_j \\ &= -\sum_{j=1}^{P} \sum_{k=1}^{K} \int q_j(\beta_j, \gamma_j = k; \theta) \log\{p(\beta_j, \gamma_j = k|\pi)\} d\beta \\ &= -\sum_{j=1}^{P} \left\{ \alpha_{j1} \log \pi_1 + \sum_{k=2}^{K} \alpha_{jk} \{\log \pi_t - \frac{1}{2} \log 2\pi\sigma_k^2\sigma_e^2 - \frac{1}{2\sigma_k^2\sigma_e^2} (\mu_{jk}^2 + s_{jk})\} \right\} \end{aligned}$$

For the third term,

$$
\begin{aligned}
\text{(D.1.12)} \quad &= \iint q(\beta,\gamma;\theta) log\{q(\beta,\gamma;\theta)\} d\beta d\gamma \\
&= \sum_{j=1}^{P}\sum_{k=2}^{K} \int q_j(\beta_j,\gamma_j=k;\theta) log\{q_j(\beta_j,\gamma_j=k;\theta)\} d\beta \\
&= \sum_{j=1}^{P} \left\{ \alpha_{j1}\log\alpha_{j1} + \sum_{k=2}^{K} \alpha_{jk}\{log\alpha_{jk} - \frac{1}{2}log2\pi s_{jk} - \frac{1}{2}\} \right\}
\end{aligned}
$$

To specify the parameters in the approximation in order to get the best fully-factorized approximation of mixture of normal, we need to find the value of the parameters that minimizes the K-L distance between posterior and approximation.

To minimize the K-L distance, we apply Lagrangian method which is a common way to optimize this type of optimization problem with some constraints on parameters. In this problem, there is only one constraint on the parameters α_{jk} for each j which is $\sum_k \alpha_{jk} = 1$. The optimization can be simplified by plugging the constraint into the objective function and then take derivative, which is equivalent to taking derivative to the Lagrangian function with Lagrangian multiplier λ_j for each j. Here

we take the simplified version which provides us an easier way to implement.

$$\begin{aligned}
\frac{\partial F}{\partial s_{jk}} &= \frac{1}{2}\alpha_{jk}(\frac{d_j}{\sigma_e^2} + \frac{1}{\sigma_k^2\sigma_e^2} - \frac{1}{s_{jk}}) \\
\frac{\partial F}{\partial \mu_{jk}} &= \alpha_{jk}(-\frac{\phi_j}{\sigma_e^2} - \frac{d_j r_j}{\sigma_e^2} + \frac{d_j\mu_{jk}}{\sigma_e^2} + \frac{\mu_{jk}}{\sigma_k^2\sigma_e^2} + \frac{1}{\sigma_e^2}\sum_{i=1}^{P}\sum_{t=1}^{P} x_{ij}x_{it}r_t) \\
\frac{\partial F}{\partial \alpha_{jk}} &= \frac{\mu_{jk}}{\sigma_e^2}(-\phi_j - d_j r_j + \sum_{i=1}^{P}\sum_{t=1}^{P} x_{ij}x_{it}r_t) + \frac{d_j}{2\sigma_e^2}(\mu_{jk}^2 + s_{jk}) \\
&\quad + log\frac{\alpha_{jk}}{\pi_k} - \frac{1}{2}log\frac{s_{jk}}{\sigma_k^2\sigma_e^2} - \frac{1}{2} + \frac{1}{2\sigma_k^2\sigma_e^2}(\mu_{jk}^2 + s_{jk}) + 1
\end{aligned}$$

for $j = 1, \cdots, P$, $k = 2, \cdots, K$ and $\sum_{k=1}^{K} \alpha_{jk} = 1; \forall j$.

D.1.2 Parameters update

Then we can get the update of parameters by setting the first order condition of objective function to zero. The unconstrained version of updates are as follows:

$$s_{jk} = \frac{\sigma_e^2}{(\mathbf{X}^T\mathbf{X})_{jj} + \frac{1}{\sigma_k^2}} \tag{D.1.14}$$

$$\mu_{jk} = \frac{s_{jk}}{\sigma_e^2}\left\{(\mathbf{X}^T y)_j - \sum_{t\neq j}(\mathbf{X}^T\mathbf{X})_{jt}r_k\right\} \tag{D.1.15}$$

$$\alpha_{jk} \propto \sqrt{\frac{s_{jk}}{\sigma_k^2\sigma_e^2}}\pi_k \exp\{\frac{1}{2}\frac{\mu_{jk}^2}{s_{jk}}\} \tag{D.1.16}$$

$$\alpha_{j1} \propto \pi_1 \tag{D.1.17}$$

After that we need to put the constrained back to the parameters as following:

$$\sum_{k=1}^{K} \alpha_{jk} = 1$$

This method only works in this certain case, since we only have a constraint on α_{jt}.

D.1.3 Hyper parameters estimation

For the hyper parameters θ, we formulate a Variational EM (VEM) approach for computing the residual variance σ_e^2 and the parameters of the mixture components. The grids of normal mixture are fixed in this case. The EM algorithm can be viewed as minimizing the K-L divergence (Neal and Hinton, 1998), or equivalently in this case, maximizing the variational lower bound of the log-likelihood. Therefore, we can obtain variational EM algorithm by computing posterior expectations in the E-step under the constraint that the true posterior is based on our fully factorized assumption. The M-step update for σ_e^2 is derived in an analogous manner to the co-ordinate ascent updates for the variational parameters, yielding

$$\sigma_e^2 = \frac{\|y - \mathbf{X}r\|^2 + \sum_j (\mathbf{X}^T\mathbf{X})_{jj} E_q(\beta_j - r_j)^2 + \sum_j \sum_{k=2}^{K} \alpha_{jk}(s_{jk}^2 + \mu_{jk}^2)/\sigma_k^2}{n + \sum_j \sum_{k=2}^{K} \alpha_{jk}}. \tag{D.1.18}$$

Similarly, the approximate M-step update for $\pi = \{\pi_1, \ldots, \pi_K\}$ is given by

$$\pi_k \propto \sum_{j=1}^{p} \alpha_{jk} \tag{D.1.19}$$

such that $\sum_k \pi_k = 1$.

Here, we can also introduce a penalty term which serves to "regularize" the estimate of π when we do not have a lot of data. This penalty term is based on the Dirichlet distribution with parameters $\lambda_k \geq 1$, and reduces to the above M-step estimator when all λ_k are equal to 1.

$$\pi_k \propto \sum_{j=1}^{p} \alpha_{jk} + \lambda_k - 1$$

such that $\sum_k \pi_k = 1$.