THE UNIVERSITY OF CHICAGO

BAYESIAN SHRINKAGE METHODS FOR HIGH-DIMENSIONAL REGRESSION

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

YOUNGSEOK KIM

CHICAGO, ILLINOIS

JUNE 2020

Copyright © 2020 by Youngseok Kim

All Rights Reserved

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This dissertation would have not been possible without the support of many people I have met during the 5 years at the University of Chicago. I dedicate this page to each and everyone who helped me complete my PhD.

First and foremost, I would like to express my sincere gratitude to my PhD advisors, Chao Gao and Matthew Stephens, for their invaluable guidance and encouragement throughout the course of this work. I extend my profound gratitude to my dissertation committee, Mihai Anitescu. Their vast academic interest, unwavering passion, solid insight and broad knowledge always enlightened, inspired and motivated me, as a lighthouse of my lifelong journey to be grown as an individual scientific researcher. Their suggestion and criticism always served as the momentum towards the lighthouse when my vision was blurred and my passion was distracted. In fact, I was fortunate enough to have the 3 shining lighthouses. It was such a wonderful experience.

I am truly grateful for all of the faculty and the staff in the Department of Statistics, for their continuous supports and endeavors.

I wholeheartedly appreciate my fellow graduate students in the Department of Statistics and all members of Stephens Lab and HELIOS Group for sharing their everyday life with me. It has been a great pleasure for me to work with these friendly, creative and talented people. I would like to express my special thanks to Peter Carbonetto, as a representative of all of my academic peers, for being such a wonderful collaborator and mentor of this work.

Lastly, I would love to thank family – my parents, Dongho Kim and Kyungmi Lee, my wife Hyemin Yoo, and my sister Youngwon Kim – for their continuous love and support.

There are many others whom I may have inadvertently left out, but I sincerely thank all of them.

ABSTRACT

High dimensional data is prevalent in modern and contemporary science, and many statistics and machine learning problems can be framed as high dimensional regression – predicting response variables and selecting relevant features among candidates. One of the most commonly used Bayesian approaches is based on the i.i.d. two component mixture prior, which comprises a point mass component at 0 ("spike") and a nonzero component ("slab"), on the regression coefficients. By computing the posterior probability that coefficients are zero, these Bayesian approaches allows us to measure the amount of shrinkage we need for individual regression coefficients and to identify relevant subsets of predictors. However, the above posterior inference can be done only approximately, and approximate inference procedures such as MCMC does not scale well to high dimensional and large data sets.

In this dissertation, we primarily focus on developing reliable Bayesian inferential tools that scale efficiently to dimensionality. Our first work proposes novel Variational Empirical Bayes (VEB) approaches to multiple linear regression based on a flexible scale mixture of normal distributions. The proposed approach (called Mr.ASH) is not only an approximate posterior inference procedure, but also a penalized regression approach where flexible EB replaces expensive cross validation. Our second work generalizes the two component mixture prior to the graph Laplacian prior, which accounts for graph structured sparsity. The general framework for Bayesian models, including sparse linear regression, change-point detection, clustering and more complex linear models with graph structured sparsity, will be presented. Our third work develops a fast algorithm for estimating mixture proportions, which serves as a central algorithmic routine in empirical Bayesian approaches to the normal means model and their applications such as linear regression and matrix factorization.

CHAPTER 1

INTRODUCTION

1.1 Perspectives on High-dimensional Regression

Recent advances in science and technology have made a breakthrough in collecting and storing large information from the world. Proliferation of massive data sets has revolutionized statistical theories and methodologies/machine learning techniques. A series of works has formed an important line of researches called *high dimensional statistics*, an area of attempting to understand data with a specific property characterized by *high dimensionality*, where the number of features (attributes which ought to explain the samples) far exceeds the number of samples (response variables subject to be predicted) collected.

High dimensional data is prevalent in modern and contemporary science, and many statistics and machine learning problems face challenges in predicting response variables from massive features and selecting relevant features among those massive candidates. Statistical modeling is one of the mainstream approaches for understanding and simulating how complex data sets are likely to be generated from the real world. The statistical model simplifies and emulates the real-world data generating process, based on the underlying statistical assumptions. The model will be learned from the data, usually by estimating the model parameters.

Statistical procedures for high dimensional data can be evaluated in many aspects. In particular, two important aspects are prediction and variable selection. The former involves predicting the unobserved future once the model is learned, and the latter involves understanding the contributions of individual features to prediction of the response variables. Achieving both accurately and simultaneously is typically not possible for high dimensional data, since prediction accuracy is often compromised by the parsimony of the model. Optimal prediction will not be usually achieved by a single parsimonious model without some form of model averaging [129].

Bayesian approaches are well-suited for these two particular aspects, since the full posterior naturally provides answers to the two primary goals: prediction and variable selection. That

is, both predicting the unobserved variables and identifying relevant subsets of features can be done by computing the posterior predictive distributions and the posterior inclusion probabilities, respectively. However, (fully) Bayesian approaches pose another challenge: posterior computation is computationally intractable in general since it involves Bayesian model averaging over exponentially many possible models. Although the intractability of the posterior precludes exact Bayesian inference, approximate procedures have been proposed with the assistance of Markov Chain Monte Carlo (MCMC) methods [128] or variational methods [83], which have become pervasive in Bayesian statistics.

Variational Bayes (VB) methods yield an estimate of the full posterior by optimizing an approximate posterior over a class of distributions for which it is easier to do inference. The quality of the approximation is usually measured by the Kullback-Leibler divergence. Compared to MCMC, variational methods tend to be faster and easier to scale to gigantic data in practice. In this thesis we focus on developing variational Bayes inference procedures that

- scale efficiently with the dimensionality;
- achieve prediction performance and/or variable selection accuracy on the acceptable level.

Accuracy of approximation will trade-off the above two main goals.

Numerous studies of complex models have emerged from fundamental studies of the multiple linear regression model. In high dimensional settings, the goal is to find a linear predictor for a n-dimensional response vector y in terms of a linear combinations of candidate predictors $\mathbf{x}_1, \cdots, \mathbf{x}_p$. An important line of research is based on penalized linear regression, solving regularized least squares problems such as [75, 151, 108, 180, 44, 176]. Also, Bayesian approaches [55, 117, 107, 28, 27, 37] have been extensively studied in parallel, developing different priors (e.g. spike-and-slab priors) on the regression coefficients and different algorithms (e.g. MCMC and VB). Despite considerable work, linear regression still remains an active research area as a cornerstone of many fields.

Penalized regression attempts to minimize the penalized log-likelihood function, hence is naturally understood as Maximum A Posteriori (MAP) estimation where the prior on the regression

coefficients serves as the penalty. By optimizing the penalized likelihood function, prediction and variable selection can be simultaneously performed, hoping the two terms (the squared loss and the penalty) correctly account for bias-variance trade-offs. The remaining part is to select the penalty by choosing parametric forms of the penalty (e.g. Lasso or L_1 penalty [151]) and by tuning parameters for the penalty. The recent few decades of abundant statistical researches in-depth ([44, 176, 47], to name a few) have witnessed asymptotic theoretical properties and practical usefulness of penalized likelihood approaches in high dimensional settings.

On the other hand, the Bayes estimator minimizing the posterior expected L_2 loss is the posterior mean [118] while the MAP estimator (i.e. the posterior mode) maximizes the posterior probability. The posterior mean of the regression coefficients, therefore, is an optimal estimator when the prior is correctly specified. Arguably, the posterior mean is understood as a more reasonable point estimator than the posterior mode (MAP). Nonetheless, when it comes to high dimensional regression, it is impossible to derive the posterior distribution analytically and is computationally intractable to develop an efficient algorithm to find the exact posterior mean.

1.2 Chapter 2: Flexible Empirical Bayes Approach to Multiple Regression

In Chapter 2, we introduce a new Empirical Bayes (EB) approach for fitting large-scale multiple linear regression, with a particular focus on predictive performance [89]. This approach combines two key ideas: (i) the use of flexible "adaptive shrinkage" priors, which approximate any scale mixture of normal distributions using a finite mixture of normals; and (ii) the use of variational methods to estimate the prior hyper-parameters and compute approximate posterior distributions. We present a simple coordinate ascent algorithm to implement this "Variational EB" (VEB) method, and show that this algorithm can be interpreted as fitting a penalized linear regression in which the form of the penalty function is learned from the data. The flexible priors, and correspondingly flexible implied penalty function, can capture a wide range of different scenarios, from very sparse to very dense regression coefficients. And yet, estimating this highly flexible prior from the data by VEB is approximately as fast as methods that penalized regression methods that tune a single tuning

parameter by cross-validation (e.g. the lasso). By adapting the prior to the data the VEB method predicts consistently well across different scenarios, typically performing as well as the best existing method for each scenario.

1.3 Chapter 3: Bayesian Variable Selection for Complex Models

Our next goal to develop of a unified framework – a basic structure of statistical procedures including models, methods and algorithms – for numerous Bayesian models. Our work stems from the study of linear regression, and covers change-point detection, clustering and many other models in the end.

In Chapter 3, we propose a general algorithmic framework for Bayesian model selection [88]. A spike-and-slab Laplacian prior is introduced to model the underlying structural assumption. Using the notion of effective resistance, we derive an EM-type algorithm with closed-form iterations to efficiently explore possible candidates for Bayesian model selection. The deterministic nature of the proposed algorithm makes it more scalable to large-scale and high-dimensional data sets compared with existing stochastic search algorithms. When applied to sparse linear regression, our framework recovers the EMVS algorithm [130] as a special case. We also discuss extensions of our framework using tools from graph algebra to incorporate complex Bayesian models such as biclustering and submatrix localization.

Many interesting Bayesian models can be framed as linear regression models with the graph-structured model parameters. To study Bayesian models from this unified perspective, we introduce a spike-and-slab Laplacian prior distribution on the model parameters, as an extension of the classical spike-and-slab prior [109, 55, 56] for Bayesian variable selection. Then the problem of Bayesian model selection can be recast as selecting a most promising subgraph from the base graph. Various choices of base graphs lead to specific statistical estimation problems such as sparse linear regression, clustering and change-point detection. In addition, the connection to graph algebra further allows us to build prior distributions for even more complicated models.

We will derive a variational EM algorithm that efficiently explores possible candidates of structural parameters. Our variational EM approach builds on top of previous work on variable

selection in linear regression [130]. The general framework proposed in this chapter can be viewed as an algorithmic counterpart of the theoretical framework for Bayesian high-dimensional structured linear models in [53].

1.4 Chapter 4: Maximum Likelihood Estimation of Mixture Proportions

Maximum likelihood estimation of mixture proportions has a long history, and continues to play an important role in modern statistics, including in development of nonparametric empirical Bayes methods. Maximum likelihood of mixture proportions has traditionally been solved using the expectation maximization (EM) algorithm, but recent work by Koenker & Mizera shows that modern convex optimization techniques—in particular, interior point methods—are substantially faster and more accurate than EM.

In Chapter 4, we develop a new solution based on sequential quadratic programming (SQP) [87]. It is substantially faster than the interior point method, and just as accurate. Our approach combines several ideas: first, it solves a reformulation of the original problem; second, it uses an SQP approach to make the best use of the expensive gradient and Hessian computations; third, the SQP iterations are implemented using an active set method to exploit the sparse nature of the quadratic subproblems; fourth, it uses accurate low-rank approximations for more efficient gradient and Hessian computations. We illustrate the benefits of the SQP approach in experiments on synthetic data sets and a large genetic association data set. In large data sets ($n \approx 10^6$ observations, $m \approx 10^3$ mixture components), our implementation achieves at least 100-fold reduction in runtime compared with a state-of-the-art interior point solver. Our methods are implemented in Julia and in an R package available on CRAN.

1.5 Summary of Accomplishments

The main contribution of the thesis is three-fold.

First of all, the proposed approaches can be framed as part of a broad research program that

seeks scalable Bayesian inference for high dimensional regression. Chapter 2 [89] focuses on the prediction accuracy in multiple linear regression, which serves as a fundamental problem in statistics and machine learning. Chapter 3 [88] attempts to extend a Bayesian variable selection tool in multiple linear regression to a model selection framework in highly structured linear regression, such as change-point regression, biclustering, isotonic regression and so forth. [53] describes a theoretical counterpart for Bayesian structured linear models.

Second, the proposed approaches address computational challenges in Bayesian high dimensional regression problems. For reliable prediction performance the prior should be flexible enough to cover a wide range of signal shapes but this flexibility increases computational burden in calculating the posterior. Chapter 2 [89] proposes a VEB approach to tackle this computational challenge. Next, Bayesian model selection in high dimensional regression chooses a best one among the exponentially many candidate models, which is impractical without some form of approximation. Chapter 3 [88] proposes a novel Bayesian approach based on a carefully designed prior on the graph. Lastly, Chapter 4 [87] develops a fast algorithm for estimating the mixture proportions, which frequently appears in compound decision theory work, as well as Empirical Bayes literature (normal means problem [144], linear regression, matrix factorization [164]).

Third, we provide the accompanying packages in R and Julia programming language. The R package `mr.ash.alpha` for Chapter 2, the Julia package `BayesMSG` for Chapter 3 and the R package `mixsqp` for Chapter 4 available online for public use.

CHAPTER 2

A FAST AND FLEXIBLE EMPIRICAL BAYES APPROACH FOR PREDICTION IN LINEAR REGRESSION

2.1 Introduction

Multiple linear regression is one of the oldest statistical methods for relating an outcome variable to predictor variables, dating back at least to the eighteenth century [e.g., 146]. In recent decades, data sets have grown rapidly in size, with the number of predictor variables often exceeding the number of observations. Fitting even simple models such as multiple linear regression to large data sets raises interesting research challenges; among these challenges, a key question is how to estimate parameters to avoid overfitting. A great variety of approaches have been proposed, including approaches based on penalized least-squares criteria [e.g., 75, 151, 44, 108, 180, 176, 73], and many Bayesian approaches [e.g., 109, 55, 106, 117, 28, 65, 66, 27, 179, 162]. The approaches differ in the choice of penalty function or prior distribution for the regression coefficients, and in the algorithm used to arrive at estimates of the coefficients. Despite considerable past work, fitting multiple linear regression models remains an active research area.

The many different approaches to this problem naturally have strengths and weaknesses. For example, ridge regression [L_2 penalty; 75, 156] has the advantages of simplicity, and competitive prediction accuracy in "dense" settings that involve many predictors with non-zero effects. However, it is not well-adapted to "sparse" settings, where a small number of non-zero predictors dominate. The Lasso [L_1 penalty; 151] is also computationally convenient, involving a simple convex optimization problem and a single tuning parameter, and behaves better in sparse settings than ridge regression. However, prediction accuracy of the Lasso is limited by its tendency to overshrink large effects [e.g. 147]. The Elastic Net [180] combines some of the advantages of ridge regression and the Lasso — and includes both as special cases — but at the cost of an additional tuning parameter, limiting its

1. This work is in collaboration with Wei Wang, Peter Carbonetto and Matthew Stephens, and builds upon the preliminary work by the second author Wei Wang [163]. This work will be submitted to a journal for peer review: Kim et. al., A Fast and Flexible Empirical Bayes Approach for Prediction in Multiple Regression, 2020.

application to large data sets. Non-convex penalties — examples include the L_0 penalty [108, 73], SCAD [44] and MCP [176] — can also give better prediction performance in sparse settings, but this comes with the cost of solving a non-convex optimization problem. Bayesian methods, by using flexible priors, have the potential to achieve excellent prediction accuracy in sparse and dense settings [e.g., 65, 175, 179], but their practical drawbacks are well-known; model fitting typically involves a Markov chain Monte Carlo (MCMC) scheme with high computational burden, and convergence of the Markov chain can be difficult to diagnose, particularly for non-expert users. In summary, when choosing among existing methods, one must confront trade-offs between prediction accuracy, flexibility, and computational convenience.

In this paper, we develop an approach to multiple linear regression that aims to combine the best features of existing methods: it is fast, comparable in speed to the cross-validated Lasso; it is flexible, capable of adapting to sparse and dense settings; it is self-tuning, with no need for user-specified hyper-parameters; and, in our numerical studies, its prediction accuracy is competitive with the best methods against which we compared, and in regression settings ranging from very sparse to very dense. Furthermore, we show that our method has a dual interpretation as either a penalized regression method or as a Bayesian regression method, thereby providing a conceptual bridge between these two distinct approaches.

Our method is based on an empirical Bayes approach to multiple regression, which assumes that the regression coefficients are independently drawn from some prior distribution that is to be estimated as part of the fitting procedure. Empirical Bayes is, in many ways, a natural candidate for attempting to attain the benefits of Bayesian methods while reducing some of their computational challenges. However, previous EB work on multiple regression has either focused on relatively inflexible priors, or has been met with considerable computational challenges. For example, [114] develop an EB approach with a normal prior — effectively EB Ridge regression — which makes computations easy, but is not well adapted to sparse settings. In contrast [54] consider the point-normal prior (sometimes called a "spike-and-slab" prior), which is considerably more flexible, but makes computation difficult. [54] make several approximations, including use of "conditional maximum

likelihood" (CML), which conditions on a single best model (i.e. which predictors have non-zero coefficients) instead of summing over all models as a conventional likelihood would. However, even the CML approximation is intractable, because finding the best model is intractable, and further approximations are required. Building on this work, [174] also uses the CML approximation to perform EB estimation for spike and slab priors, but they replace the normal slab with a Laplace (double Exponential) slab. They then improve computation by reducing flexibility: specifically, they constrain the two parameters of this prior in a way that creates interesting connections with the Lasso, which they then exploit to make the CML computation more tractable.

Here we take a different EB approach that is both more flexible than previous EB approaches and also computationally scalable. This new approach has two key components. First, to increase flexibility we exploit the "adaptive shrinkage" priors used in [144]; specifically we use the scale mixture of normals version of these priors. This prior family includes most of popular choices of the priors that have been used in Bayesian regression, including Normal, Laplace, Point-Normal, Point-Laplace, Point-t, Normal-Inverse-Gamma and Horseshoe [75, 56, 106, 107, 66] priors. Notably, while increasing flexibility usually comes at the computational expense, in this case the use of the adaptive shrinkage priors actually simplifies many computations, essentially because the scale mixture family is a convex family. Second, to make computations tractable, we exploit the variational approximation (VA) methods for multiple regression from [27]. Although these VA methods are approximations, they improve on the CML approximation because they sum over a large set of plausible models rather than simply selecting a single best model. The main limitation of this VA approach is that, in sparse settings with very highly correlated predictors, it will give only one of the correlated predictors a non-negligible coefficient [27]. This limitation, which is shared by the Lasso and L_0-penalized regression as they tend to select only one of the highly correlated predictors, does not greatly affect prediction accuracy, but does mean that other approaches [e.g. 162] may be preferred if the goal is to do variable selection for scientific interpretation, rather than for prediction.

We call this variational approach to EB "Variational Empirical Bayes" (VEB). Although motivated by Bayesian ideas, the VEB approach has close connections with penalized linear regression

(PLR) methods. Indeed, the VEB approach is, like PLR methods, is based on solving an optimization problem. Furthermore, the algorithm we use to do this is, at its heart, a coordinate-ascent algorithm for fitting a PLR. However, whereas existing PLR methods assume a relatively restrictive class of penalty functions, usually with just one or two parameters that are tuned by cross-validation, our VEB approach uses a much more flexible family of penalty functions (corresponding to our flexible family of priors), and the form of the penalty function is itself learned from the data (by solving an optimization problem, analogous to how EB learns priors from data by maximizing the likelihood). While one might expect such a flexible approach to substantially increase computational complexity, in fact the VEB approach has a similar computational burden to methods that tune a single parameter by CV (e.g. Lasso), and is substantially faster than tuning two parameters by CV (e.g. Elastic Net). Our methods are implemented as an R package, `mr.ash.alpha` ("Multiple Regression with Adaptive SHrinkage priors," alpha release), available at `https://github.com/stephenslab/mr.ash.alpha`.

2.1.1 Organization of the Paper

The remainder of this paper is organized as follows. Section 2.2 provides backgrounds and preliminaries, and outlines our approach. Section 2.3 illustrates the VEB methods and the model-fitting algorithms. Section 2.4 draws connections between our approach and penalized regression. Section 2.5 provides numerical studies to compare prediction performance of our VEB methodss with existing methods. Section 2.6 summarizes the contributions, and discusses future directions.

2.1.2 Notations and Conventions

We write vectors in bold, e.g., $\mathbf{b}$, and matrices in bold capital letters, e.g., $\mathbf{X}$. We write sets and families in calligraphic font, e.g., $\mathcal{G}$. We use $\mathcal{N}(\,\cdot\,; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ to denote the probability density function of the multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Our convention is to reserve i for row indices and j for column indices of a matrix, so that $\mathbf{x}_i$ and $\mathbf{x}_j$ denote, respectively, the i-th row and j-th column of matrix $\mathbf{X}$. Finally, $\|\mathbf{x}\| = (\mathbf{x}^T\mathbf{x})^{1/2}$ denotes the Euclidean, or L_2, norm of $\mathbf{x}$.

2.2 Outline and Preliminaries

2.2.1 Empirical Bayes Linear Regression

In this paper, we develop empirical Bayes methods [42, 126] to fit a basic multiple linear regression model,

$$\mathbf{y} \,|\, \mathbf{X}, \mathbf{b}, \sigma^2 \sim \mathcal{N}(\,\cdot\,; \mathbf{X}\mathbf{b}, \sigma^2 I_n), \tag{2.1}$$

where $\mathbf{y} \in \mathbb{R}^n$ is a vector of responses; $\mathbf{X} \in \mathbb{R}^{n \times p}$ is a design matrix whose columns contain predictors $\mathbf{x}_1, \ldots, \mathbf{x}_p \in \mathbb{R}^n$; $\mathbf{b} \in \mathbb{R}^p$ is a vector of regression coefficients; σ^2 is the variance of the residual errors; and $\mathbf{I}_n$ is the $n \times n$ identity matrix. While an intercept is not explicitly included in (2.1), it is easily accounted for by centering $\mathbf{y}$ and the columns of $\mathbf{X}$ prior to model fitting; see also Section 2.3.3. To simplify the presentation, we will assume throughout the paper that the columns of $\mathbf{X}$ are rescaled so that $\|\mathbf{x}_j\|^2 = 1$, for $j = 1, \cdots, p$. (A special case of this assumption is that the columns of $\mathbf{X}$ are "standardized.") Note that this rescaling is not required for the actual implementation of the proposed methods; In the Appendix 2.7.2, we provide more general descriptions of the algorithm and generalizes the theoretical results to the unscaled, or unstandardized, case.

We assume a prior distribution in which the scaled regression coefficients, b_j/σ, are independent and identically distributed (*i.i.d.*) from some distribution with density g. In other words,

$$b_j \,|\, g, \sigma^2 \overset{iid}{\sim} g_\sigma(\,\cdot\,), \tag{2.2}$$

where $g_\sigma(b_j) \triangleq g(b_j/\sigma)/\sigma$ is σ-scaled prior on the regression coefficients. (We formulate the prior in terms of the scaled coefficients because it leads to some algorithmic simplifications later; however all our methods can also be applied, with minor modifications, to work with the unscaled prior $b_j \,|\, g, \sigma^2 \overset{iid}{\sim} g(\,\cdot\,)$.) Although our methods apply more generally, we focus on priors g that are scale mixtures of normals — this is a computationally convenient choice that preserves flexibility.

Specifically, following [144], we assume $g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ where

$$\mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2) \triangleq \left\{ g = \sum_{k=1}^{K} \pi_k \mathcal{N}(\,\cdot\,; 0, \sigma_k^2) : \pi \in \mathcal{S} \right\}, \quad \mathcal{S} \triangleq \left\{ \pi \in \mathbb{R}_+^K : \sum_{k=1}^{K} \pi_k = 1 \right\}. \quad (2.3)$$

Here, $0 \leq \sigma_1^2 < \cdots < \sigma_K^2 < \infty$ is a pre-specified grid of component variances and $\pi_1, \cdots, \pi_K$ are unknown mixture proportions. Typically the first variance σ_1^2 would be set exactly to zero to allow for a sparse regression model (here we adopt the convention that $\mathcal{N}(\,\cdot\,; 0, 0)$ is point mass at zero, δ_0). By making this grid of variances sufficiently wide and dense, the prior family $\mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ can approximate, with arbitrary accuracy, the nonparametric family of all the scale mixtures of zero-mean normal distributions. This nonparametric family, which we denote $\mathcal{G}_{\text{SMN}}$, includes most popular distributions used as priors in Bayesian regression models, motivated by different applications, including Normal ("Ridge regression") [75], Point-Normal ("spike and slab") [56, 109], Double Exponential or Laplace ("Bayesian Lasso") [117], Horseshoe [28], the Normal-Inverse-Gamma prior [107, 66], mixture of two normals (BSLMM) [179], and the mixture of four zero-centered normals with different variances (BayesR) suggested by [111].

To simplify derivations, it is helpful to think of any $g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ as being parameterized by its unknown mixture proportions $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$ and known component variances $\sigma_1^2, \ldots, \sigma_K^2$. In some cases, it is also convenient to express the normal mixture prior (2.2), i.e.

$$b_j \,|\, g, \sigma^2 \overset{iid}{\sim} \sum_{k=1}^{K} \pi_k \mathcal{N}(\,\cdot\,; 0, \sigma^2 \sigma_k^2) \quad (2.4)$$

using a standard augmented-variable representation:

$$\begin{aligned} \Pr(\gamma_j = k \,|\, g) &= \pi_k, \\ b_j \,|\, g, \gamma_j = k &\sim \mathcal{N}(\,\cdot\,; 0, \sigma^2 \sigma_k^2), \end{aligned} \quad (2.5)$$

where the discrete latent variable $\gamma_j \in \{1, \ldots, K\}$ indicates which mixture component gave rise to b_j.

A basic EB approach to fitting the regression model (2.1, 2.2) would involve the following two steps:

1. Estimate the prior density g and the error variance σ^2 by maximizing the marginal likelihood:

$$\begin{aligned}(\widehat{g}, \widehat{\sigma}^2) &= \underset{g \in \mathcal{G}, \sigma^2 \in \mathbb{R}_+}{\operatorname{argmax}} \; p(\mathbf{y} \,|\, \mathbf{X}, g, \sigma^2) \\ &= \underset{g \in \mathcal{G}, \sigma^2 \in \mathbb{R}_+}{\operatorname{argmax}} \; \log \int p(\mathbf{y} \,|\, \mathbf{X}, \mathbf{b}, \sigma^2)\, p(\mathbf{b} \,|\, g, \sigma^2)\, d\mathbf{b}.\end{aligned} \tag{2.6}$$

2. Perform inference for $\mathbf{b}$ based on its posterior distribution,

$$p_{\text{post}}(\mathbf{b} \,|\, \mathbf{y}, \mathbf{X}, \widehat{g}, \widehat{\sigma}^2) \propto p(\mathbf{y} \,|\, \mathbf{X}, \mathbf{b}, \widehat{\sigma}^2)\, p(\mathbf{b} \,|\, \widehat{g}, \widehat{\sigma}^2). \tag{2.7}$$

   In particular, one typically estimates b_j by its posterior mean, $\mathbb{E}(b_j \,|\, \mathbf{X}, \mathbf{y}, \widehat{g}, \widehat{\sigma}^2) = \mathbb{E}_{p_{\text{post}}}(b_j)$ for prediction.

The combination of maximization (Step 1) and computing posterior expectations (Step 2) suggests an expectation maximization (EM) approach [34], but, unfortunately, both steps are computationally intractable, except in special cases. For Step 1 (parameter estimation), the optimization is hard due to the intractable integral. For Step 2 (posterior inference), computation of the posterior (2.7) is intractable due to the normalization constant.

Our strategy to circumvent the intractability of these steps is to approximate the intractable posteriors using a mean-field variational approximation [17, 83, 159]. The resulting coordinate ascent algorithm [18, 58, 133] or CAVI [17, 123], and is grounded by the generalized EM framework introduced in [112].[1] Mean-field variational approximations have previously been used to implement empirical Bayes methods with intractable posteriors [18, 161, 165], and for fitting Bayesian linear regression models [6, 27, 61, 99, 162], but not for linear regression with the flexible class of priors (2.3) we consider here. Effectively, our work here combines this previous work by [27] with recent work in [144] that takes an EB approach to a simpler problem – the "normal means" problem

1. Note that this is not the same as the "Variational Bayesian EM" algorithm described by [10].

[43, 80, 148] — with the flexible priors (2.3). The result is methods that are simpler, faster, more flexible, and generally more accurate than those in [27]. Before describing our approach in detail, we briefly review the key ideas that provide the building blocks for our approach.

2.2.2 The Empirical Bayes Normal Means Model

The Normal Means (NM) model is a model for a sequence $y_1, \cdots, y_n$ of observations, each of which is normally distributed with unknown mean b_j and known variance σ^2:

$$y_j \,|\, b_j, \sigma^2 \sim \mathcal{N}(\,\cdot\,; b_j, \sigma^2), \quad j = 1, \cdots, n. \tag{2.8}$$

This can be viewed as a special case of the multiple regression model (2.1) with $\mathbf{X} = \mathbf{I}_n$ (the $n \times n$ identity matrix) and known residual variance σ^2. [144] provides methods to fit the model (2.8,2.2), by Empirical Bayes (i.e. Steps 1 and 2 above, but with σ^2 fixed) for various classes of prior, with a focus on the scale mixtures of normals (2.3). In what follows, we will present an EB approach for the NM model with this particular choice (2.3) of the prior family $\mathcal{G}$.

Step 1, estimating the prior g, is made particularly simple by the use of a fixed grid of variances in (2.3), which means that only the mixture proportions $\boldsymbol{\pi}$ need to be estimated. This is easily done by maximizing the (marginal) likelihood $p(\mathbf{y} \,|\, g, \sigma^2)$. Indeed, exploiting the latent variable representation (2.5), maximum likelihood estimation of π can be written as an optimization problem:

$$\widehat{\boldsymbol{\pi}} = \operatorname*{argmax}_{\boldsymbol{\pi} \in \mathcal{S}} \sum_{j=1}^{p} \log \sum_{k=1}^{K} \pi_k L_{jk} \tag{2.9}$$

where each L_{jk} is each component (marginal) likelihood, i.e.

$$L_{jk} \triangleq p(y_j \,|\, g, \sigma^2, \gamma_j = k) = \mathcal{N}(y_j; 0, \sigma^2(1 + \sigma_k^2)). \tag{2.10}$$

The optimization of (2.9) is a convex optimization problem, which can be solved efficiently using convex optimization methods [91, 87], or more simply by iterating the following Expectation

Maximization (EM) algorithm [34] until convergence:

$$\begin{aligned}
&\texttt{E-step}: \ \phi_{jk} \leftarrow \phi_k(y_j; g, \sigma^2) \triangleq \Pr(\gamma_j = k \,|\, y_j, g, \sigma^2) = \frac{\pi_k L_{jk}}{\sum_{l=1}^{K} \pi_l L_{jl}}, \quad j = 1, \cdots, p, \\
&\texttt{M-step}: \ \pi_k \leftarrow \frac{1}{p} \sum_{j=1}^{p} \phi_{jk}, \quad k = 1, \cdots, K.
\end{aligned} \tag{2.11}$$

The posterior component assignment probabilities ϕ_{jk} are sometimes referred to as the "responsibilities" (e.g. Hastie et al. [72]).

Step 2, computing the posteriors, is analytically tractable, essentially because of the independence of the observations and the conjugacy of the normal (mixture) prior to the normal likelihood. Specifically:

$$\begin{aligned}
p_{\text{post}}^{\text{NM}}(b_j, \gamma_j = k \,|\, y_j, g, \sigma^2) &= \Pr(\gamma_j = k \,|\, y, g, \sigma^2)\, p(b_j \,|\, y, g, \sigma^2, \gamma_j = k) \\
&= \phi_k(y_j; g, \sigma^2)\, \mathcal{N}\big(b_j; \mu_k(y_j; g, \sigma^2), \sigma^2 \sigma_k^2/(1+\sigma_k^2)\big),
\end{aligned} \tag{2.12}$$

where each component posterior mean is

$$\mu_k(y; g, \sigma^2) \triangleq y \sigma_k^2/(1+\sigma_k^2). \tag{2.13}$$

Summing the component posterior (2.12) over k gives the following analytic expression of the posterior for b_j:

$$p_{\text{post}}^{\text{NM}}(b_j \,|\, y_j, g, \sigma^2) = \sum_{k=1}^{K} \phi_k(y_j; g, \sigma^2)\, \mathcal{N}\big(b_j; \mu_k(y_j; g, \sigma^2), \sigma^2 \sigma_k^2/(1+\sigma_k^2)\big). \tag{2.14}$$

2.2.3 Variational Inference for Linear Regression

We will use the Variational Approximation (VA) to perform approximate Bayesian inference for multiple linear regression. The key idea of VA is to approximate the intractable posterior distribution,

$$p_{\text{post}}(\mathbf{b}, \boldsymbol{\gamma}) \triangleq p(\mathbf{b}, \boldsymbol{\gamma} \,|\, \mathbf{y}, \mathbf{X}, g, \sigma^2) \tag{2.15}$$

by a best approximation (e.g. in terms of the Kullback-Leibler divergence) q of p_{post} in some family $\mathcal{Q}$ that is chosen to make computations tractable. A common strategy is to take $\mathcal{Q}$ to be a set of distributions that factorize in some way [16, 37, 123]. In particular, [27] follow this strategy with the family:

$$\mathcal{Q} = \left\{ q : q(\mathbf{b}, \boldsymbol{\gamma}) = \prod_{j=1}^{p} q_j(b_j, \gamma_j) \right\}. \tag{2.16}$$

(They also assume a specific parametric form of each variational factor q_j, but this is unnecessary and we do not assume this here. The parametric form of each q_j will be automatically determined as its optimal form when it is updated by the algorithms we present later.)

The VA then seeks

$$\begin{aligned} \widehat{q}(g, \sigma^2) &\triangleq \operatorname*{argmin}_{q \in \mathcal{Q}} D_{KL}(q \,\|\, p_{\text{post}}) \\ &= \operatorname*{argmin}_{q \in \mathcal{Q}} \int q(\mathbf{b}, \boldsymbol{\gamma}) \log \frac{q(\mathbf{b}, \boldsymbol{\gamma})}{p_{\text{post}}(\mathbf{b}, \boldsymbol{\gamma})} d\mathbf{b} \end{aligned} \tag{2.17}$$

where $D_{KL}(q \,\|\, p)$ denotes the Kullback-Leibler (KL) divergence from a distribution q to a distribution p [92], and the notation $\widehat{q}(g, \sigma^2)$ makes explicit the fact that $\widehat{q}$ depends on g, σ^2 (because p_{post} does). In words, the VA attempts to find a best approximate posterior $\widehat{q}(g, \sigma^2)$ within the variational family $\mathcal{Q}$ by miminizing the KL divergence.

Because $D_{KL}(q \,\|\, p_{\text{post}})$ is itself computationally intractable, the optimization problem (2.17) is usually recast as the equivalent problem [83, 16, 17]

$$\widehat{q}(g, \sigma^2) \triangleq \operatorname*{argmax}_{q \in \mathcal{Q}} F(q, g, \sigma^2), \tag{2.18}$$

where

$$F(q, g, \sigma^2) \triangleq \log p(\mathbf{y}|\mathbf{X}, g, \sigma^2) - D_{KL}(q \,\|\, p_{\text{post}}) \tag{2.19}$$

is called the "Evidence Lower Bound" (ELBO) because it is a lower-bound for the "evidence" $\log p(\mathbf{y}|\mathbf{X}, g, \sigma^2)$, since the D_{KL} term is always non-negative. Although each of the two terms

on the right hand side of (2.19) are intractable, some cancellations occur that make F itself more computationally tractable. [27] provide an algorithm to find $\widehat{q}$ when the prior g is a point-normal distribution – that is, $g(\cdot) = \pi_0\delta_0(\cdot) + (1-\pi_0)\mathcal{N}(\cdot; 0, \sigma_b^2)$. This can be viewed as a special case our approach, with $K = 2$ and $\sigma_1^2 = 0; \sigma_2^2 = \sigma_b^2$, although they estimate σ_b^2 rather than fixing it.

To deal with the fact that g, σ^2 are unknown, Carbonetto and Stephens [27] perform approximate Bayesian inference for these parameters (equivalently, for $\pi_0, \sigma_b^2, \sigma^2$). They do this by treating $F(\widehat{q}, g, \sigma^2)$ as a direct approximation to the evidence. That is, they use the approximation

$$p(\mathbf{y}|\mathbf{X}, g, \sigma^2) \approx \exp\left(F(\widehat{q}(g, \sigma^2), g, \sigma^2)\right), \tag{2.20}$$

and combine this approximate likelihood with a prior on g, σ^2 to compute approximate posterior distributions. Carbonetto and Stephens [27] perform these posterior computations approximately using importance sampling, which they found to perform well in their numeric studies; however, it is computationally burdensome because it requires many optimizations of (2.18).

2.3 Variational Empirical Bayes Multiple Linear Regression

With these preliminaries in place, our approach is straightforward to describe. Essentially we combine the EB methods from [144] for the normal means problem and the VA from [27] for the multiple regression problem, to produce a new simple approach based on EB and VA. Specifically we solve the following optimization problem:

$$\text{maximize} \quad F(q, g, \sigma^2) \quad \text{subject to} \quad q \in \mathcal{Q}, \;\; g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2), \;\; \sigma^2 \in \mathbb{R}_+. \tag{2.21}$$

and obtain the solution $(\widehat{q}, \widehat{g}, \widehat{\sigma}^2)$.

Step 1 (2.6) and Step 2 (2.7) are recast into the single optimization problem (2.21). The solution $\widehat{q}$ serves as an approximate posterior for Step 2, and $(\widehat{g}, \widehat{\sigma}^2)$ serves as an approximate maximum likelihood estimate for Step 1.

Algorithm 1 Coordinate Ascent algorithm for maximizing F (Outline).

Initialize: $q_1, \cdots, q_p, g, \sigma^2$.
repeat
  for $j = 1, \cdots, p$ **do**
    $q_j \leftarrow \text{argmax}_{q_j} F(q_j, q_{-j}, g, \sigma^2)$.
  end for
  $g \leftarrow \text{argmax}_{g \in \mathcal{G}} F(q, g, \sigma^2)$.
  $\sigma^2 \leftarrow \text{argmax}_{\sigma^2 \in \mathbb{R}_+} F(q, g, \sigma^2)$.
until convergence criteria met.
Output: $(\widehat{q}, \widehat{g}, \widehat{\sigma}^2) = (q, g, \sigma^2)$.

This approach is closely related to the variational approach in [27], but replaces the point-normal prior with the more flexible scale mixture family $\mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ and their numerical integration over g and σ^2 with a maximization over g and σ^2 (as in conventional in EB methods). These modifications lead to substantial algorithmic simplifications, and computational speed-up, without sacrificing prediction accuracy (see Section 2.5).

We call this approach Variational Empirical Bayes (VEB). Note that the standard EB procedure corresponds exactly to solving the optimization problem (2.21) with no constraint on q [162]. Thus the difference between EB and VEB is the addition of the constraint $q \in \mathcal{Q}$, which turns the problem from an intractable problem into a tractable problem.

2.3.1 A Coordinate Ascent Algorithm

To solve the optimization problem (2.21) we use a coordinate ascent (CA) algorithm, which optimizes F with respect to each of its parameters in turn, at each step keeping the other parameters constant (Algorithm 1).

The key result of this section – and, effectively, the key to the attractiveness of the VEB approach – is that each step of Algorithm 1 involves only simple analytic computations from the Empirical Bayes normal means model (Section 2.2.2). In particular, the update for q_j involves computing a posterior distribution under a normal means problem (see (2.14)), and the update for g involves running one step of the EM algorithm (see (2.11)) for the Empirical Bayes normal means model

(2.8, 2.2). These results are summarised in the following Proposition:

Proposition 2.3.1 (Coordinate Ascent Algorithm)**.** *Suppose* $\mathbf{x}_j^T \mathbf{x}_j = 1$. *Let* q_{-j} *denote all elements of* q *except* q_j. *Let* $\bar{\mathbf{r}}_j$ *denote the expectation, under* q_{-j}*, of the residuals computed using all variables except* j*:*

$$\bar{\mathbf{r}}_j \triangleq \mathbf{y} - \mathbf{X}_{-j}\bar{\mathbf{b}}_{-j} \in \mathbb{R}^n, \tag{2.22}$$

where $\mathbf{X}_{-j}$ *denote the matrix* $\mathbf{X}$ *excluding the* j*th column, and* $\bar{\mathbf{b}}_{-j}$ *denotes the expectations (i.e. first moments) of* q_{-j}. *Let* $\widetilde{b}_j$ *denote the univariate ordinary least squares (OLS) estimator of* b_j *when regressing* $\bar{\mathbf{r}}_j$ *against* $\mathbf{x}_j$*:*

$$\widetilde{b}_j \triangleq \mathbf{x}_j^T \bar{\mathbf{r}}_j \in \mathbb{R}. \tag{2.23}$$

Then

1. *The optimum* $\widehat{q}_j \triangleq \operatorname{argmax}_{q_j} F(q_j, q_{-j}, g, \sigma^2)$ *is achieved by the posterior distribution* $p^{\mathrm{NM}}_{\mathrm{post}}(\cdot|\widetilde{b}_j, g, \sigma^2)$ (2.14) *for* b_j, γ_j *under the normal means model (2.8, 2.2), with data given by* $\widetilde{b}_j$ *and variance* σ^2*:*

$$\widehat{q}_j = p^{\mathrm{NM}}_{\mathrm{post}}(\,\cdot\,|\,\widetilde{b}_j, g, \sigma^2). \tag{2.24}$$

   In particular, $\widehat{q}_j$ *is a mixture of normals, with component mixture proportions* $\phi_{jk} = \phi_k(\widetilde{b}_j; g, \sigma^2)$ *and component means* $\mu_{jk} = \mu_k(\widetilde{b}_j; g, \sigma^2)$.

2. *The optimum* $\widehat{g} \triangleq \operatorname{argmax}_{g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)} F(q, g, \sigma^2)$ *is achieved by*

$$\widehat{g} = \sum_{k=1}^{K} \widehat{\pi}_k g_k, \quad \textit{where} \quad \widehat{\pi}_k = (1/p)\sum_{j=1}^{n} \phi_{jk} \ \textit{for} \ k = 1, \cdots, K. \tag{2.25}$$

3. *The optimum* $\widehat{\sigma}^2 \triangleq \operatorname{argmax}_{\sigma^2 \in \mathbb{R}_+} F(q, g, \sigma^2)$ *is achieved by:*

$$\widehat{\sigma}^2 = \frac{1}{n + p(1-\pi_1)}\left[\bar{\mathbf{r}}^T\bar{\mathbf{r}} + \sum_{j=1}^{p}(\widetilde{b}_j - \bar{b}_j)\bar{b}_j + p(1-\pi_1)\sigma^2_{\mathrm{old}}\right], \tag{2.26}$$

   where $\bar{\mathbf{r}} \triangleq \mathbf{y} - \mathbf{X}\bar{\mathbf{b}} \in \mathbb{R}^p$ *is the expected residual vector and* σ^2_{old} *is the previous iterate of* σ^2.

Algorithm 2 Coordinate Ascent algorithm for maximizing F (Detailed).

Initialize: $\bar{\mathbf{b}}, \boldsymbol{\pi}, \sigma^2$
$\bar{\mathbf{r}} = \mathbf{y} - \mathbf{X}\bar{\mathbf{b}}$ (compute residuals)
repeat
  for $j = 1, \cdots, p$ **do**
    $\widetilde{b}_j \leftarrow \bar{b}_j + \mathbf{x}_j^T \bar{\mathbf{r}}$. (compute OLS estimate)
    for $k = 1, \cdots, K$ **do**
      $\phi_{jk} \leftarrow \phi_k(\widetilde{b}_j; g, \sigma^2)$ (compute $\widehat{q}_j$)
      $\mu_{jk} \leftarrow \mu_k(\widetilde{b}_j; g, \sigma^2)$
    end for
    $\bar{b}_j \leftarrow \sum_k \phi_{jk}\mu_{jk}$. (compute posterior mean)
    $\bar{\mathbf{r}} \leftarrow \mathbf{y} - \mathbf{X}\bar{\mathbf{b}}$. (done efficiently without recomputing $\mathbf{X}\bar{b}$)
  end for
  for $k = 1, \cdots, K$ **do**
    $\pi_k \leftarrow p^{-1} \sum_{j=1}^p \phi_{jk}$. (compute $\widehat{g}$)
  end for
  $\sigma^2 \leftarrow n^{-1}\left[\|\bar{\mathbf{r}}\|^2 + \sum_{j=1}^p (\widetilde{b}_j - \bar{b}_j)\bar{b}_j\right]$. (estimate residual variance)
until convergence criteria met.
Output: $(\widehat{\mathbf{b}}, \widehat{\boldsymbol{\pi}}, \widehat{\sigma}^2) = (\bar{\mathbf{b}}, \boldsymbol{\pi}, \sigma^2)$ (one could also output $\boldsymbol{\phi}, \boldsymbol{\mu}$ if desired)

Proof. See Appendix 2.7.2. □

Inserting these results into Algorithm 1 gives Algorithm 2. Because this algorithm is based on the exact coordinate ascent that sequentially updates each variable by its coordinate maximizer, every iteration is guaranteed to increase F. The algorithm is also guaranteed to converge, as summarized in the following Proposition (analogous to Proposition 1 of Breheny and Huang [23], which establishes convergence of coordinate descent for SCAD and MCP penalties):

Proposition 2.3.2 (Convergence)**.** *The sequence* $\{q^{(t)}, g^{(t)}, (\sigma^2)^{(t)}\}$ *for* $t = 1, 2, \ldots$ *generated by Algorithm 2 converges monotonically to a local maximum of* F.

Proof. See Appendix 2.7.4. □

We have written Algorithm 2 to mirror Algorithm 1, but in practice we rearrange computations slightly to reduce run-time and memory requirements. Also, we do not need to assume $\mathbf{x}_j^T \mathbf{x}_j = 1$

in practice; it was assumed for simplifying the discussion. See Algorithm 4 in Appendix 2.7.2 for actual implementation details.

This implementation is scalable, with computational complexity $O((n+K)p)$ per iteration (i.e. each single cycle of updating $q_1, \cdots, q_p, g$ and σ^2), and memory requirements $O(n+p+K)$ (in addition to $O(np)$ required to store $\mathbf{X}$, which could be avoided by an implementation that holds only a subset of the columns of the $\mathbf{X}$ matrix in memory at any time). Note that to achieve these memory requirements one must avoid storing the $O(Kp)$ variational parameters (i.e. only stores $\bar{\mathbf{b}}$ and $\bar{\mathbf{r}}$).

2.3.2 Accuracy of VEB; Exactness for Orthogonal Predictors

[27] note that their VA approach provides the exact posterior distribution when the columns of $\mathbf{X}$ are orthogonal. The following is a natural extension of this result, showing that in this special case our VEB approach is exactly the usual EB approach.

Proposition 2.3.3. *When* $\mathbf{X}$ *has orthogonal columns solving the VEB problem* (2.21) *is equivalent to the usual EB two-step procedure (2.6, 2.7).*

Proof. See Appendix 2.7.4. □

This result follows from the fact that, when $\mathbf{X}$ has orthogonal columns, the posterior distribution for $\mathbf{b}$ does indeed factorize as (2.16), and thus the mean field "approximation" is actually exact. That is, $\widehat{q} = \arg\max_{q \in \mathcal{Q}} F(q, g, \sigma^2)$ is equal to the true posterior p_{post}, and therefore the ELBO $\max_{q \in \mathcal{Q}} F(q, g, \sigma^2)$ is equal to the marginal log-likelihood $\log p(\mathbf{y}|\mathbf{X}, g, \sigma^2)$ for g, σ^2. We note in passing that the CML approach to approximate EB inference, used in [54, 174], is not exact even in the case of orthogonal columns.

Proposition 2.3.3 suggests that our VEB method will be accurate when $\mathbf{b}$ is sparse and the columns of $\mathbf{X}$ are close to orthogonal; e.g. when the columns are approximately independent. It also suggests that the approximation may be less accurate for very highly correlated columns. However, [27] discuss this issue in depth, and explain why, although the VA for $\mathbf{b}$ is inaccurate for highly correlated columns, the estimated hyper-parameters (e.g., here the prior g) are nonetheless accurate.

They also note that, when two predictors are very highly correlated with one another, and also predictive of $\mathbf{y}$, the VA tends to give just one of them an appreciable estimated coefficient. This is similar to the behaviour of many PLR methods (including Lasso), but different from most MCMC methods (at convergence). While this behaviour is undesirable when interest focuses on variable selection for scientific interpretation, it does not adversely affect prediction accuracy [162]. Thus, for prediction, the VEB approach may be expected to perform well even in settings with correlated predictors. Our numerical studies (Section 2.5) confirm this.

2.3.3 Practical Issues and Extensions

The VEB method (Algorithm 2 solving (2.21)) with our specific choice $\mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ of the prior family will be called Mr.ASH ("Multiple Regression with Adaptive SHrinkage priors"). In what follows, our practical considerations for the implementation of Mr.ASH will be provided.

Intercept: In multiple regression applications it is common to include an intercept term that is not regularized in the same way as other variables. This is usually achieved by centering both $\mathbf{y}$ and the columns of $\mathbf{X}$, by subtracting their means respectively, before fitting the regression model [48]. After fitting, the intercept term is estimated by $\bar{\mathbf{y}} - \sum_{j=1}^{p} \bar{\mathbf{x}}_j \bar{\mathbf{b}}_j$ and is added to the prediction. This is the strategy we take here.

Selection of grid for prior variances: As in [144], we suggest choosing a grid $\{\sigma_1^2, \cdots, \sigma_K^2\}$ that is sufficiently broad and dense that results do not change much if the grid is made broader and denser. The goal here is that $\mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ be made to approximate $\mathcal{G}_{\mathrm{SMN}}$. We choose the lower end of the grid to be $\sigma_1^2 = 0$, corresponding to point mass at 0. We choose the upper end $\sigma_K^2 \approx n$, so that the prior variance of $\mathbf{x}_j b_j$ is close to σ^2 at most (recall that we assumed $\mathbf{x}_j^T \mathbf{x}_j = 1$, hence $\mathrm{Var}(\mathbf{x}_j) \approx 1/n$ when $\mathbf{x}_j$ is centered). We have found a grid of 20 points spanning this range to be sufficient to achieve reliable prediction performance across many examples (see Section 2.5). Thus our default grid is $K = 20$ and $\sigma_k^2 = n(2^{(k-1)/20} - 1)^2$.

In principle, when $\text{Var}(\mathbf{x}_j b_j)$ is much larger than σ^2 for some j, then we need a larger σ_K^2 not to underestimate the effect size of b_j by over-shrinking it. To this end, our software checks that the final estimated weight π_K on the largest grid point is negligible (i.e. less than convergence tolerance), and emits a warning if not, since this suggest that the grid may need to be made wider.

Initialization: Maximizing F is generally a non-convex optimization problem, and so although Algorithm 2 is guaranteed to converge, the solution obtained may depend on initialization of $\bar{\mathbf{b}}$, $\boldsymbol{\pi}$ and σ^2. The simplest initialization for $\bar{\mathbf{b}}$ is $\bar{b}_j = 0$ for all j, which we call the "null" initialization. Another initialization, used by [177] for example, is to initialize $\bar{\mathbf{b}}$ to the solution from the lasso (after cross-validation), which we call $\widehat{\mathbf{b}}^{\text{lasso}}$. Given $\bar{\mathbf{b}}$ we initialize the algorithm with $\sigma^2 = (1/n)(\mathbf{y} - \mathbf{X}\bar{\mathbf{b}})^T(\mathbf{y} - \mathbf{X}\bar{\mathbf{b}})$. We also initialize $\boldsymbol{\pi} = (1/K, \cdots, 1/K)$.

In numerical studies (Section 2.5) we find that initialization to $\widehat{\mathbf{b}}^{\text{lasso}}$ can perform appreciably better than the null initialization in settings where columns of $\mathbf{X}$ are highly correlated. In other settings the two initializations perform similarly.

Convergence criterion: We terminate Algorithm 2 at iteration t if $\|\boldsymbol{\pi}^{(t)} - \boldsymbol{\pi}^{(t-1)}\|_\infty = \max_k |\pi_k^{(t)} - \pi_k^{(t-1)}|$ is less than a convergence tolerance, whose default setting is 10^{-8}.

Extension to other mixture prior families: Although $\mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ is our focus of the paper, Algorithm 2 can be modified to perform VEB with other mixture prior families $\mathcal{G} \triangleq \{g = \sum_{k=1}^K \pi_k g_k : \boldsymbol{\pi} \in \mathcal{S}\}$, with fixed mixture component distributions $g_1, \ldots, g_K$. The only requirements are (i) the convolution of each g_k with a normal likelihood is numerically tractable, and (ii) the posterior mean can be computed for the normal means model with prior $b_j \sim g_k$. For example, this holds if g_k are point masses, or uniform or Laplace distributions.

However, computations for normal mixtures are particularly simple, fast and numerically stable. Further, the family of scale mixtures of normals includes most prior distributions previously used for multiple regression, so we believe these will suffice for most practical applications.

Table 2.1: The parametric formulation of the penalty functions and their shrinkage operators.

Method & Name	Penalty Function $\rho(t)$	Shrinkage Operator $S(t)$
Normal shrinkage (Ridge, L_2)	$\frac{\lambda}{2}t^2$	$\frac{t^2}{1+\lambda}$
Soft-thresholding (Lasso, L_1)	$\lambda\|t\|$	$0, \quad \text{if } \|t\| \le \lambda$ $t-\lambda, \quad \text{if } \lambda < t$ $t+\lambda, \quad \text{if } b < -\lambda$
Elastic Net	$(1-\eta)\frac{\lambda}{2}t^2 + \eta\lambda\|t\|$	$S_{\text{soft},\eta\lambda/a}(t/a)$, $a = 1+(1-\eta)\lambda$.
Hard-thresholding (Best subset, L_0)	$\lambda\mathbb{I}\{\|t\| > 0\}$	$0, \quad \text{if } \|t\| \le \lambda$ $t, \quad \text{if } \lambda < t$ $t, \quad \text{if } b < -\lambda$
Smoothly Clipped Absolute Deviation	$\lambda\|t\|, \quad \text{if } \|t\| \le 2\lambda$ $\frac{\eta\lambda\|t\| - 0.5(t^2+\lambda^2)}{\eta-1}, \quad \text{if } 2\lambda < \|t\| \le \eta\lambda$ $\frac{\lambda^2(\eta+1)}{2}, \quad \text{if } \eta\lambda < \|t\|$	$S_{\text{soft},\lambda}(t), \quad \text{if } \|t\| \le 2\lambda$ $\frac{S_{\text{soft},\eta\lambda/(\eta-1)}(t)}{1-(1/(\eta-1))}, \quad \text{if } 2\lambda < \|t\| \le \eta\lambda$ $t, \quad \text{if } \eta\lambda < \|t\|$
Minimax Concave Penalty	$\lambda\|t\| - \frac{t^2}{2\eta}, \quad \text{if } \|t\| \le \eta\lambda$ $\frac{\eta\lambda^2}{2}, \quad \text{if } \eta\lambda < \|t\|$	$\frac{S_{\text{soft},\lambda}(t)}{1-(1/(\eta-1))}, \quad \text{if } \|t\| \le \eta\lambda$ $t, \quad \text{if } \eta\lambda < \|t\|$

2.4 Variational Empirical Bayes and Penalized Linear Regression

The VEB approach outlined above is closely related to the penalized linear regression (PLR) methods, which fit the multiple regression model by minimizing the regularized squared loss function:

$$\text{minimize}_{\mathbf{b}\in\mathbb{R}^p} \quad h_\rho(\mathbf{b}) \triangleq \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{b}\|^2 + \sum_{j=1}^{p}\rho(b_j), \tag{2.27}$$

for some penalty function ρ. Some common choices for the penalty function are given in Table 2.1, including L_2 (Ridge; Hoerl and Kennard [75]), L_0 (Miller [108]), L_1 (Lasso; Tibshirani [151]), Elastic Net (E-NET; Zou and Hastie [180]), Smoothly Clipped Absolute Deviation (SCAD; Fan and Li [44]) and Minimax Concave Penalty (MCP; Zhang [176]). In this section we explain the

connection between the VEB approach and penalized linear regression.

The PLR problem (2.27) is often tackled using the coordinate descent algorithm [e.g. 48, 169, 23, 73]; that is, by iterating over the coordinates of b sequentially, at each step solving (2.27) for one coordinate while keeping the other coordinates fixed. Assuming the columns of X are standardized, i.e. $\mathbf{x}_j^T \mathbf{x}_j = 1$, the solution for the j-th coordinate is achieved by the update:

$$b_j \leftarrow S_\rho \left(b_j + \mathbf{x}_j^T (\mathbf{y} - \mathbf{X}\mathbf{b}) \right), \quad j = 1, \cdots, p, \tag{2.28}$$

where

$$S_\rho(y) \triangleq \operatorname*{argmin}_{b \in \mathbb{R}} \frac{1}{2}(y - b)^2 + \rho(b), \tag{2.29}$$

is known as the shrinkage/thresholding operator corresponding to penalty ρ (also known as the univariate proximal operator with respect to ρ; [116]). For commonly-used penalty functions, the corresponding shrinkage operators are analytically available; see Table 2.1.

To connect the VEB approach to PLR methods, we initially restrict our attention to a special case of the VEB approach (Algorithm 2) where g and σ^2 are fixed, rather than estimated, so only q is estimated. The key computation in the inner loop of Algorithm 2, which maximizes F over each q_j in turn, is to compute the posterior mean $\bar{b}_j$. (When (g, σ^2) is known and fixed, the other computations in the inner loop – computing ϕ_{jk} and μ_{jk} – are required only to compute $\bar{b}_j$.) Further, from Proposition 2.3.1, this posterior mean is computed under a simple normal means model. It is therefore helpful to introduce notation for this key computation:

Definition 2.4.1 (Normal Means Posterior Mean Operator)**.** *For prior g and variance σ^2, define the "Normal means posterior mean operator", $S_{g,\sigma^2} : \mathbb{R} \to \mathbb{R}$, to be the mapping*

$$S_{g,\sigma^2}(y) \triangleq \mathbb{E}_{\mathrm{NM}}(b_j \mid y_j = y, g, \sigma^2) \tag{2.30}$$

where $\mathbb{E}_{\mathrm{NM}}$ denotes expectation under the normal means model (2.8,2.2).

Algorithm 3 Coordinate Ascent Iterative Shrinkage Algorithm for Variational Posterior Mean (fixed g, σ^2)

Initialize: $\bar{\mathrm{b}}$
repeat
  for $j = 1, \cdots, p$ **do**
    $\bar{b}_j \leftarrow S_{g,\sigma^2}(\bar{b}_j + \mathbf{x}_j^T(\mathbf{y} - \mathbf{X}\bar{\mathbf{b}}))$
  end for
until convergence criteria met
Output: $\widehat{\mathrm{b}} = \bar{\mathrm{b}}$

For $g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$, it follows from (2.14) that S_{g,σ^2} has a simple analytic form,

$$S_{g,\sigma^2}(y) = \sum_{k=1}^{K} \phi_k(y; g, \sigma^2)\mu_k(y; g, \sigma^2). \tag{2.31}$$

It is easy to show that the NM posterior mean operator $S_{g,\sigma}$ is an odd function, and is monotonic in y. Also, S_{g,σ^2} is a shrinkage operator, in that $|S_{g,\sigma^2}(y)| \leq |y|$. See Lemma 2.7.3 for the proof.

With this notation, we can write the inner loop of Algorithm 2 – the coordinate updates of $q_1, \cdots, q_p$ given g and σ^2 – in simplified form as Algorithm 3. This simplified algorithm has exactly the same form as the coordinate descent algorithm (2.28) for PLR, but with the shrinkage operator S_ρ in (2.28) replaced with the posterior mean operator S_{g,σ^2} (2.30). We note, in passing, that this algorithm also works for any prior g, and not only for $g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$. The similarity of the two algorithms, Algorithm 3 and (2.28), suggests that given g and σ^2 the variational posterior mean under the model (2.1, 2.2) can be understood as solving a PLR problem (2.27) for some penalty ρ depending on g and σ^2. The following Theorem formalizes this by constructing the penalty.

Theorem 2.4.1 (Variational Posterior Mean Solves Penalized Linear Regression)**.** *Let g, σ^2 be such that the inverse, $S_{g,\sigma}^{-1}$, of the shrinkage operator $S_{g,\sigma}$ (2.30) exists. Let $\widehat{\mathrm{b}}$ denote a variational posterior mean, meaning a solution output by Algorithm 3. Then $\widehat{\mathrm{b}}$ is a local minimizer of the PLR objective function h_ρ (2.27), for any penalty $\rho = \rho_{g,\sigma}$ whose derivative ρ' satisfies*

$$\rho'(b) = S_{g,\sigma}^{-1}(b) - b. \tag{2.32}$$

Furthermore, if $\widehat{q}$ is a global maximizer of the ELBO (F), then also $\widehat{\mathbf{b}}$ is a global minimizer of h_ρ.

Proof. See Appendix 2.7.4 for the proof. Note that for any penalty $\rho_{g,\sigma}$ satisfying (2.32), the corresponding shrinkage operator (2.29) is $S_{g,\sigma}$. □

Note that taking $g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ and $\sigma^2 > 0$ satisfies the conditions of Theorem 2.4.1, because the corresponding shrinkage operator (2.31) is a strictly increasing function, and so its inverse exists. Note also that adding a boundary condition $\rho(0) = 0$ ensures the existence of a unique solution to (2.32) (by the Cauchy–Lipschitz theorem).

2.4.1 Remarks: Variational Bayes as Penalized Regression

Theorem 2.4.1 establishes that, with g and σ^2 fixed, the variational Bayes (VB) approach (Algorithm 3, without implementation of EB) is solving a PLR problem for a fixed penalty. We make the following remarks.

Special case: normal prior and ridge regression: In the special case where the prior g is a single zero-mean normal distribution, the NM posterior mean shrinkage operator $S_{g,\sigma}$ exactly matches the ridge regression (or L_2) shrinkage operator (Table 2.1). Thus in this special case Algorithm 3 is solving the ridge regression problem (PLR with L_2 penalty), which we note is a convex problem. Furthermore, both are equivalent to finding the *true* posterior mean of $\mathbf{b}$ under this normal prior. This is easy to see: Ridge regression finds the true posterior mode for $\mathbf{b}$ under the normal prior, and in this special case the posterior mean is equal to the posterior mode because the posterior is multivariate normal (whose mean and mode are the same). Thus in this special case, although the variational posterior approximation q does not exactly match the true posterior (because the true posterior does not factorize as in (2.16)), the variational posterior mean *does* exactly match the true posterior mean.

Flexibility of the posterior mean shrinkage operator: The shape of the posterior mean shrinkage operator $S_{g,\sigma}$ naturally depends on the prior g. Different choices of g can lead to quite different

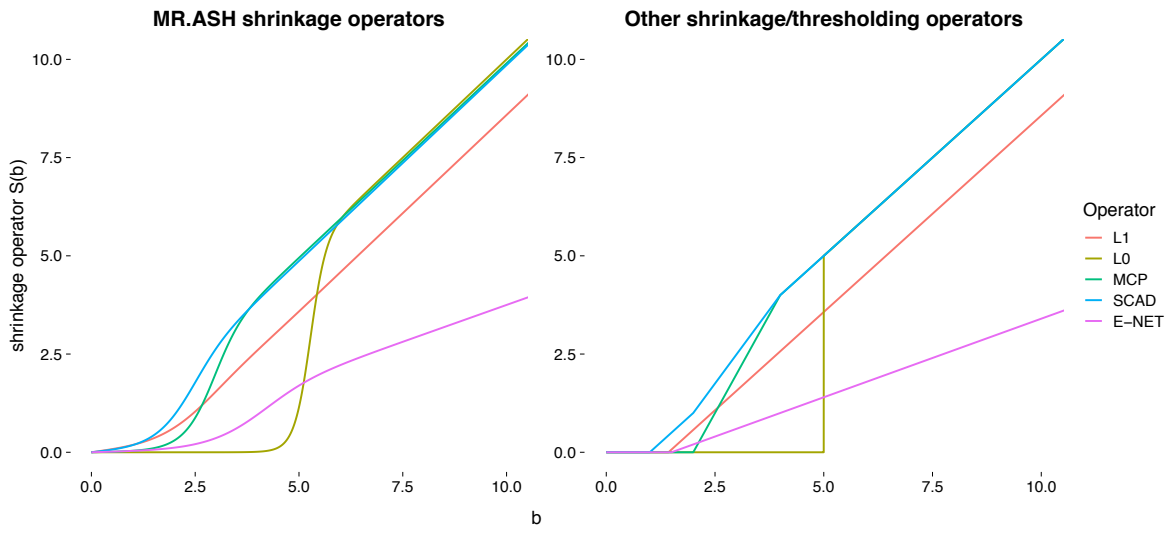


Figure 2.1: Examples of posterior mean shrinkage operators for various $g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ *(Left)*, chosen to mimic some shrinkage operators corresponding to well-known penalties *(Right)*.

shapes of shrinkage operators. Further, by suitably selecting $g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)$ we can obtain shrinkage operators that are qualitatively similar to well-known shrinkage operators of existing PLR methods; see Figure 2.1 for illustration. Thus, although only Ridge regression is formally included as a special case of our VEB approach, these results emphasise the flexibility of the VEB approach compared with existing PLR methods, and suggest potential to mimic the behaviour of a wide range of existing PLR methods. In other words, as Elastic Net includes Ridge and Lasso as special cases, Mr.ASH includes several existing PLR methods (e.g. Table 2.1) as special cases approximately or qualitatively.

Posterior mean versus posterior mode: Traditional PLR approaches are sometimes described in Bayesian terms, as obtaining the posterior mode (or *maximum a posterior*, MAP) estimate for $\mathbf{b}$ given a certain prior. For example, Lasso is the MAP under a Laplace prior. In contrast, the variational approach seeks the *posterior mean* not the posterior mode. In particular, note that the variational approach with a Laplace prior would *not* be the Lasso (it would be closer to the Bayesian Lasso of [117], which is quite different).

From a Bayesian decision theoretic perspective (e.g. Chapter 4 of Berger [11]), the posterior

mean for $\mathbf{b}$ has much better theoretical support: not only does it minimize the expected mean-squared-error for estimating $\mathbf{b}$, it also minimizes the expected mean-squared-error of predictions ($\widehat{y}_i = \mathbf{x}_i^T \mathbf{b}$). Also, the posterior mean is an admissible point estimate (e.g. Chapter 5 of [95]). In contrast, the posterior mode has very little support as an estimator for $\mathbf{b}$, particularly if prediction of $\mathbf{y}$ is the ultimate goal. For example, with any sparse prior that has non-zero mass on 0 the posterior mode is always $\mathbf{b} = \mathbf{0}$, which will generally provide poor prediction performance. (Note also that if $\widehat{\mathbf{b}}^{\text{mode}}$ denotes the posterior mode for $\mathbf{b}$ then $\mathbf{x}_i^T \widehat{\mathbf{b}}^{\text{mode}}$ is *not* generally the posterior mode of $\widehat{y}_i = \mathbf{x}_i^T \mathbf{b}$.)

2.4.2 Remarks: Estimating g, σ^2 by Empirical Bayes

We return now to the general case, where g, σ^2 are not fixed, but rather estimated by maximum likelihood (Algorithm 2). Theorem 2.4.1 implies that the variational posterior mean $\widehat{\mathbf{b}}$ obtained from Algorithm 2 is a solution to the PLR with a penalty, $\rho = \rho_{\widehat{g},\widehat{\sigma}}$, that is effectively estimated during the fitting procedure. That is, VEB is solving a PLR, with *the shape of the penalty function being estimated from the data*.

Estimating the shape of the penalty function from the data by VEB can be seen as analogous to estimating the tuning parameters in regular PLR methods, which is usually done by cross-validation (CV) [48, 23]. However our VEB approach considers a much more flexible set of penalty functions than is usually considered (Figure 2.1). Indeed, traditional PLR methods use penalty functions with just one or two tuning parameters, whereas ours is parameterized by K degrees of freedom ($\pi_1, \cdots, \pi_K$ and σ^2). CV requires to solve the PLR problem (2.27) for each tuning parameter value and each validation set, and thus tuning more than one parameter by CV is cumbersome. Therefore, tuning anything as flexible as our approach by CV seems impractical. However, the VEB approach makes it straightforward; it is self-tuning ($\boldsymbol{\pi}$ and σ^2 are estimated together with q via solving the single optimization problem (2.21)). Indeed the computational burden of the VEB approach is similar to methods that tune a single tuning parameter by CV, and much lower than methods that tune two tuning parameters (see Section 2.5.2 for empirical evidence).

2.5 Numerical Studies

2.5.1 Experiment Settings

In this section we conduct extensive numerical experiments to compare our method with other methods. We conduct a wide range of simulation settings determined by the following parameters:

n: the number of samples in training data.

p: the number of predictor variables.

$\mathbf{X}$: the design matrix of predictor variables.

s: the sparsity level (number of predictors with non-zero coefficients), which can vary from 1 (very sparse) to p (very dense).

h: the signal distribution (underlying distribution of the non-zero coefficients).

PVE: the "Proportion of variance explained", which is a number in $[0, 1]$ that summarizes the total proportion of variance in y that is explained by the predictors.

Given these parameters we generate both test and training data by first generating $\mathbf{X} \in R^{2n \times p}$ if $\mathbf{X}$ is not a given real data set; by randomly selecting s predictors to have non-zero coefficients, and generating these s non-zero regression coefficients independently from h; and by sampling the response variables $\mathbf{y}$ from $\mathbf{y} \sim \mathcal{N}(\mathbf{Xb}, \sigma^2 \mathbf{I}_{2n})$, where $\sigma^2 = \text{PVE} \cdot \text{Var}(\mathbf{Xb})$ to achieve the desired level of PVE (here Var denotes sample variance of a vector). We then randomly divide the $2n$ samples into equal-sized test and training data, $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}})$ and $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$, respectively.

For the design matrix we consider three settings:

EquicorrGauss(ρ): $\mathbf{X} \sim \mathcal{MN}(0, \mathbf{I}_{2n}, \mathbf{\Sigma}_\rho)$, where $\mathcal{MN}$ denotes the matrix normal distribution and $\mathbf{\Sigma}_\rho$ is the equi-correlated covariance matrix having unit diagonal entries and constant off-diagonal entries ρ.

IndepGauss: Equivalent to $\mathbf{X} \sim$ *EquicorrGauss(0)*, the entries of $\mathbf{X}$ are independent standard normal random variables.

RealGenotype: Human genotype data from 20 different genes, from the Genotype-Tissue Expression project [64].

The real genotypes are taken from Wang et al. [162], and exhibit complex patterns of correlations among predictors, with some pairs of predictors being very highly-correlated ($\approx \pm 1$). Each dataset contains the SNPs (genetic variants) with minor allele frequencies > 0.05 within 1 Megabase of each gene's transcription start site; see Wang et al. [162] for details. Under the *RealGenotype* scenario the values of n and p are determined directly by the $\mathbf{X}$, rather than being specified: the sample size is $2n = 574$ (so $n = 287$) and p varies from 4012 to 8760.

We define simulation scenarios by first defining a baseline simulation scenario, and then defining additional simulation scenarios by specifying only how they vary from this baseline. The baseline is:

$$\textit{Baseline}: n = 500,\ p = 2000,\ s = 20,\ \mathbf{X} \sim \textit{IndepGauss},\ \text{PVE} = 0.5,\ h = \mathcal{N}(0, 1).$$

The six additional scenarios differ from the baseline as follows:

Low-dimension : $p = 200$.

High-dimension : $p = 10000$.

PointConstant-signal : $p = 200$ and $h = \delta_1$ (point mass at 1).

Strong-signal : $p = 200$, PVE $= 0.9$.

Equicorr($\rho = 0.95$) : $\mathbf{X} \sim \textit{EquicorrGauss}(0.95)$.

RealGenotype : $\mathbf{X} \sim \textit{RealGenotype}$ (so $n = 287$ and $4012 \leq p \leq 8760$).

Table 2.2: Summary of the 11 methods compared.

Method	R Package	Brief Description	Reference
Mr.ASH	`mr.ash.alpha`	The proposed VEB method	This paper
Ridge Lasso E-Net	`glmnet`	PLR with the ridge penalty. PLR with the lasso penalty. PLR with the elastic net penalty.	Friedman et al. [48]
SCAD MCP	`ncvreg`	PLR with the scad penalty. PLR with the minimax concave penalty.	Breheny and Huang [23]
L0Learn	`L0Learn`	PLR with the L_0 penalty.	Hazimeh and Mazumder [73]
BayesB B-Lasso	`BGLR`	MCMC with $\pi_0\delta_0 + (1-\pi_0)\sigma_{\mathbf{b}}t_5$ prior on $\mathbf{b}$. MCMC with the scaled Laplace(λ) prior on $\mathbf{b}$.	Pérez and de Los Campos [120]
SuSiE	`susieR`	VB with $\mathbf{b} = \sum_{l=1}^{L} \mathbf{z}_l b_l$ and multinomial priors on $\mathbf{z}_l$'s and normal priors on b_l's.	Wang et al. [162]
varbvs	`varbvs`	VB with $\pi_0\delta_0 + (1-\pi_0)\mathcal{N}(0, \sigma_{\mathbf{b}}^2)$ prior on $\mathbf{b}$ and discrete BMA with respect to π_0.	Carbonetto and Stephens [27]

Comparing Prediction Accuracy We apply each method to the training data to obtain an estimate $\widehat{\mathbf{b}}$ for the regression coefficient. We then assess predictive performance on the test data by calculating $\widehat{\mathbf{y}}_{\text{test}} := \mathbf{X}_{\text{test}}\widehat{\mathbf{b}}$ and computing the root-mean-squared-error (RMSE) divided by σ to obtain a scale-free performance measure (Pred.Err):

$$\begin{aligned} \text{RMSE}(\mathbf{y}_{\text{test}}, \widehat{\mathbf{y}}_{\text{test}}) &\triangleq \frac{\|\mathbf{y}_{\text{test}} - \widehat{\mathbf{y}}_{\text{test}}\|}{\sqrt{n}}, \\ \text{Pred.Err}(\mathbf{y}_{\text{test}}, \widehat{\mathbf{y}}_{\text{test}}) &\triangleq \frac{\text{RMSE}(\mathbf{y}_{\text{test}}, \widehat{\mathbf{y}}_{\text{test}})}{\sigma}. \end{aligned} \tag{2.33}$$

Note that

$$1 \leq \mathbb{E}_{\boldsymbol{\epsilon}_{\text{test}}}[\text{Pred.Err}(\mathbf{y}_{\text{test}}, \widehat{\mathbf{y}}_{\text{test}})] \leq \sqrt{\frac{1}{1 - \text{PVE}}},$$

provided that $\mathbf{X}_{\text{train}}$, $\mathbf{X}_{\text{test}}$ and $\boldsymbol{\epsilon}_{\text{test}}$ are independent of one another.

Comparison methods We compare our method, Mr.ASH, with ten other methods: six penalized linear regression methods (Ridge, Lasso, E-NET, SCAD, MCP, L0Learn) and four Bayesian regression methods (BayesB, B-Lasso, SuSiE, varbvs). These methods were chosen to reflect a range of different

penalties, priors and fitting procedures (e.g. MCMC vs Variational) among methods with R package implementations. See Table 2.2 for the summary of competitors.

Each method has many ways that it could be applied. Even for Lasso one has many decisions to make, including what kind of cross-validation (CV) to use (e.g. 5-fold vs 10-fold), what criteria to use for selecting the optimal tuning parameter, and whether to relax the fit or not. With so many methods to compare we do not attempt to address the tricky issue of how best to tune each method. Instead we attempt to apply them "out-of-the-box", largely following software default settings. We note two minor exceptions to this: with E-NET we tune both tuning parameters by CV (rather than tuning just one parameter, which is the software default); and for SuSiE we set the upper limit on the number of non-zero coefficients to $L = 20$, replacing the default ($L = 10$), because most of our simulations involve more than 10 non-zero coefficients. All the R code used to perform the comparisons is available at `https://doi.org/10.5281/zenodo.3754715`.

The following remarks summarise our prior expectations about the performance of these methods:

- Ridge and B-Lasso are well adapted to dense signals, and so should perform well for that case, but may perform poorly for sparse signals.

- L0Learn and SuSiE are well adapted to sparse signals, and so should perform well for that case, but may perform poorly for dense signals.

- Lasso is a convex method that has a reputation for being fast but can suffer from bias in parameter estimates, particularly overshrinking strong signals [e.g. 147, 78].

- E-NET is also convex, and more flexible than Lasso and Ridge, including both as a special case. It may therefore perform well across a wider range of conditions at the expense of more computation time.

- SCAD and MCP are based on non-convex penalties designed to reduce the bias suffered by Lasso parameter estimates. They might therefore outperform Lasso, possibly at a cost in some additional computation. Since these methods were developed primarily for sparse regression it is unclear how well they will adapt to dense signals.

- BayesB is a Bayesian method with spike-and-slab prior (here, using a t_5 slab), which has the potential to perform well for both sparse and dense signals provided the runs are long enough for the MCMC fitting procedure to converge.

- varbvs and Mr.ASH are variational approximation methods based on the same variational approximation. The main differences are: (i) Mr.ASH uses a simple EB approach to estimate prior, whereas varbvs uses a more complex approach that, with software defaults, favors sparse models; (ii) Mr.ASH uses the Lasso initialization to improve performance with correlated predictors. We therefore expect to see Mr.ASH outperform varbvs in dense settings and cases with correlated predictors.

2.5.2 Experiment Results

We conducted three main groups of simulations aimed at assessing how the performance of different methods varies with different parameters:

Group 1 : Varying sparsity (s).

Group 2 : Varying sample size (n), signal strength (PVE), and signal shape (h).

Group 3 : Varying number of predictors (p).

We divide the simulations this way because s has a particularly strong effect on the relative prediction performance of many methods, and p has a strong effect on compute time.

For the Group 1, "varying-s", simulations we simulated data under each of the six "additional scenarios" defined above, and allowed s to vary from 1 to p. This involves four scenarios with uncorrelated designs (Group 1a) and two with correlated design matrices (Group 1b). For the Group 2 simulations we took the *Baseline* scenario, and varied each parameter n, PVE and h in turn. Because the variation in h has different effects depending on s, we did an additional simulation with h varying when the signal is dense ($p = 200$ and $s = 200$). Note that in our *Baseline* setting, the

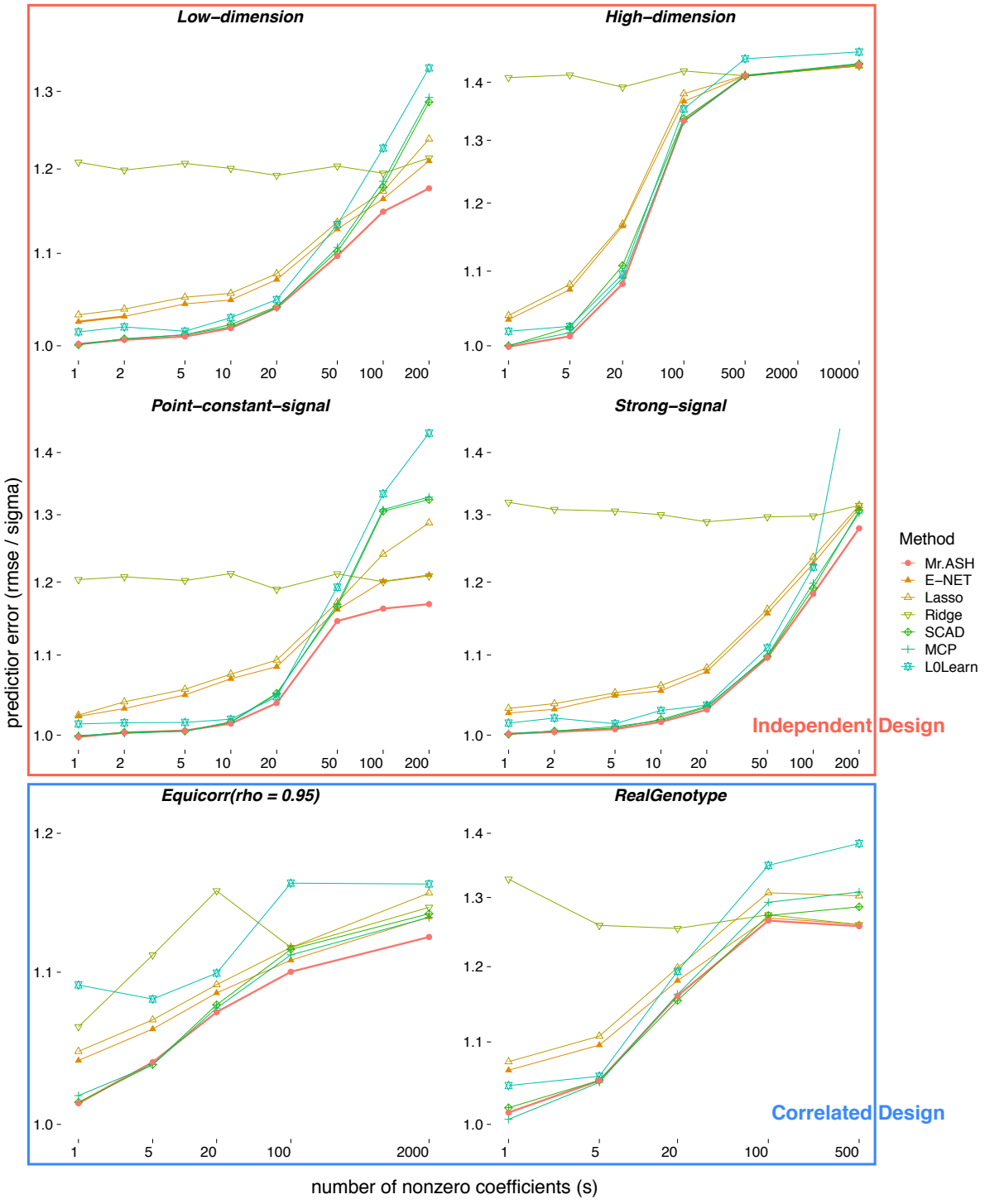


Figure 2.2: Comparison of prediction error of penalized regression methods in simulations with varying levels of sparsity s. Each panel shows a different simulation setting as s varies from $1 - p$. Each point is the average prediction error (2.33) over 20 independent simulations.

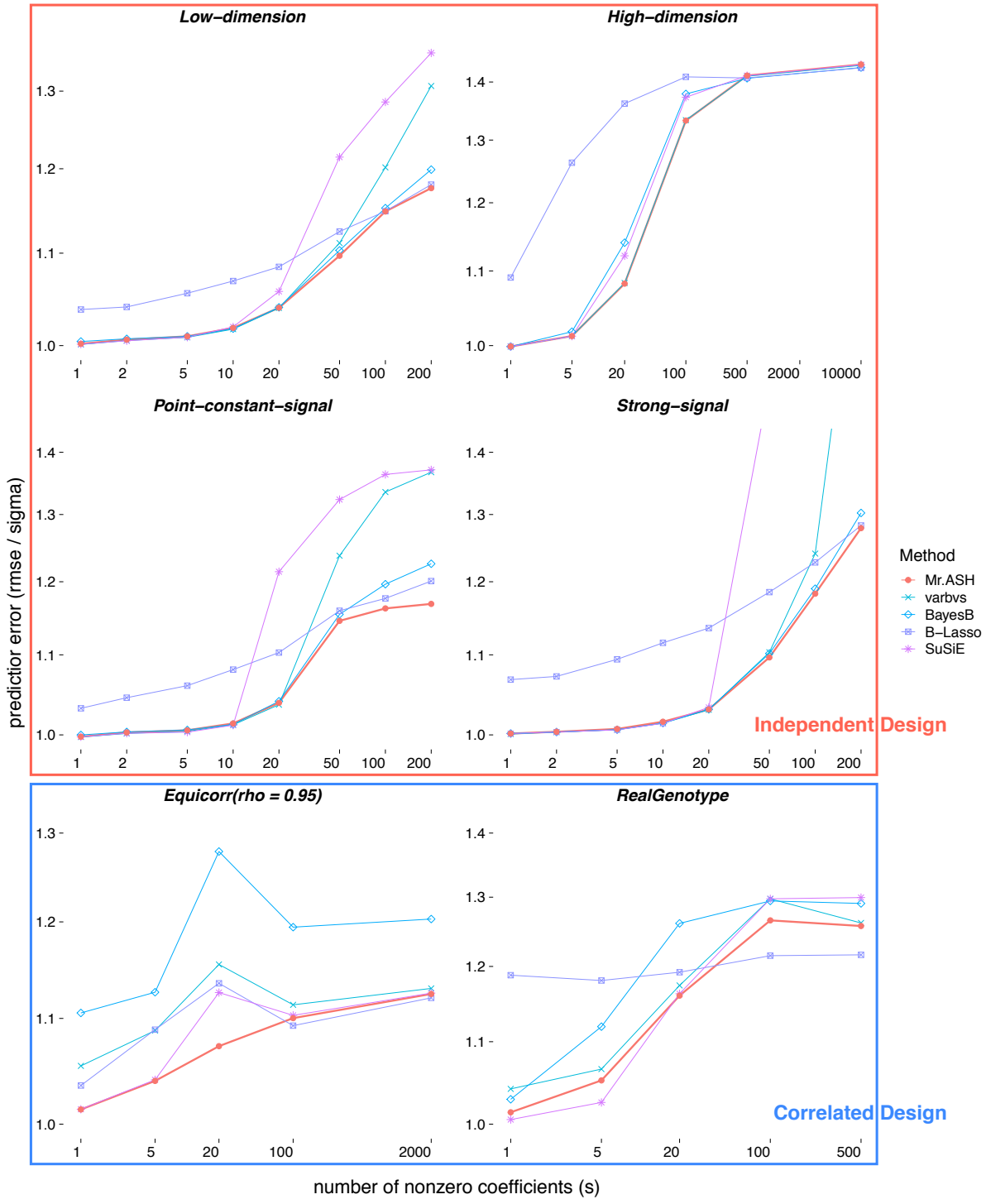


Figure 2.3: Comparison of prediction error of Bayesian regression methods in simulations with varying levels of sparsity s. Each panel shows a different simulation setting as s varies from $1 - p$. Each point is the average prediction error (2.33) over 20 independent simulations. These are the same simulations as in Figure 2.2.

signal is sparse ($p = 2{,}000$ and $s = 20$). For Group 3, "varying-p", we took the *Baseline* scenario and varied p from 20 to 20,000.

The results for varying s are shown in Figures 2.2 and 2.3; results for varying n, PVE, and h are in Figure 2.4; and results for varying p are in Figure 2.5. We summarize the main features of these results in turn.

Performance with varying sparsity level (s): The comparison of penalized regression methods (Figure 2.2) shows that, as expected, relative performance of some methods varies considerably with s. As might be expected, L0Learn, performs competitively only in sparse settings, whereas Ridge is competitive only in dense settings. The Lasso and E-NET perform more consistently across different s, with E-NET consistently slightly better, but there is a clear performance gap between these and the best method in each setting. Generally MCP and SCAD perform similarly, and often competitively, with the exception of dense signals under some scenarios (e.g. *Low-dimension*, *PointConstant-signal*). Overall Mr.ASH performs as well or better than all other methods in all scenarios.

The comparison with Bayesian methods on the same scenarios (Figure 2.3) shows some analogous patterns. Again, some methods are competitive only in sparse scenarios (SuSiE) or dense scenarios (B-Lasso). The BayesB method performs consistently poorly in the scenarios with correlated predictors, presumably because the default MCMC run-lengths are too short to ensure reliable results for these harder settings. And although varbvs is based on the same variational approximation as Mr.ASH, its performance is generally worse than Mr.ASH. This is particularly evident in dense settings – probably because the varbvs hyper-parameters are chosen to favor sparsity – and in correlated designs, probably because, unlike Mr.ASH, varbvs does not exploit warm start from the Lasso solution; see also Section 2.5.4.

Overall Mr.ASH is consistently close to the best-performing method. The main places it is outperformed are in scenarios with correlated designs, where B-Lasso is better for dense signals and SuSiE is slightly better for sparse signals. This is presumably because the variational approximation

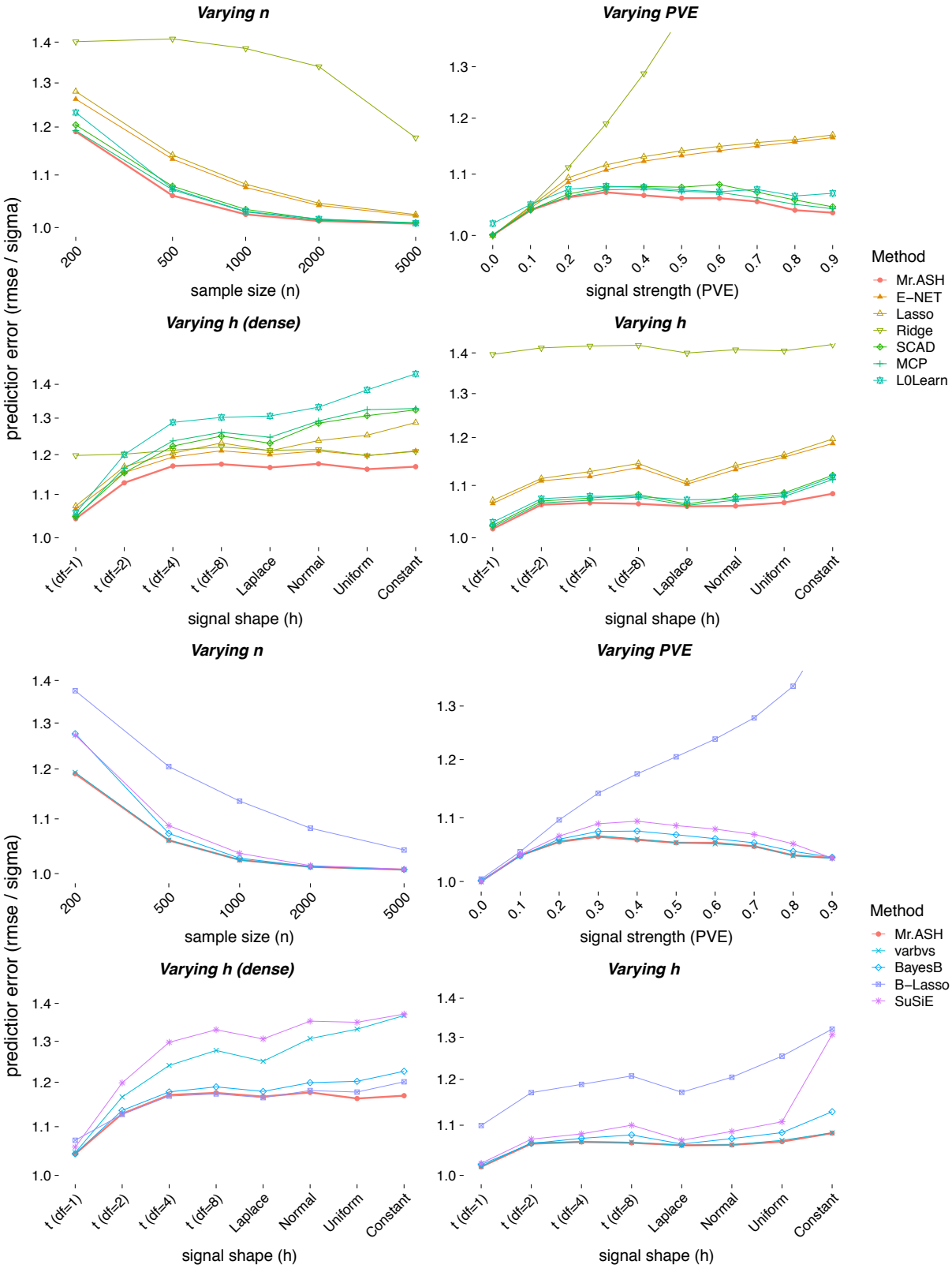


Figure 2.4: Comparison of prediction performance for varying sample sizes (n), signal strength (PVE) and signal shape (h). Each point shows average prediction error over 20 independent simulations. In the varying h simulations, the x axis is ordered from longer tail to shorter-tail distributions.

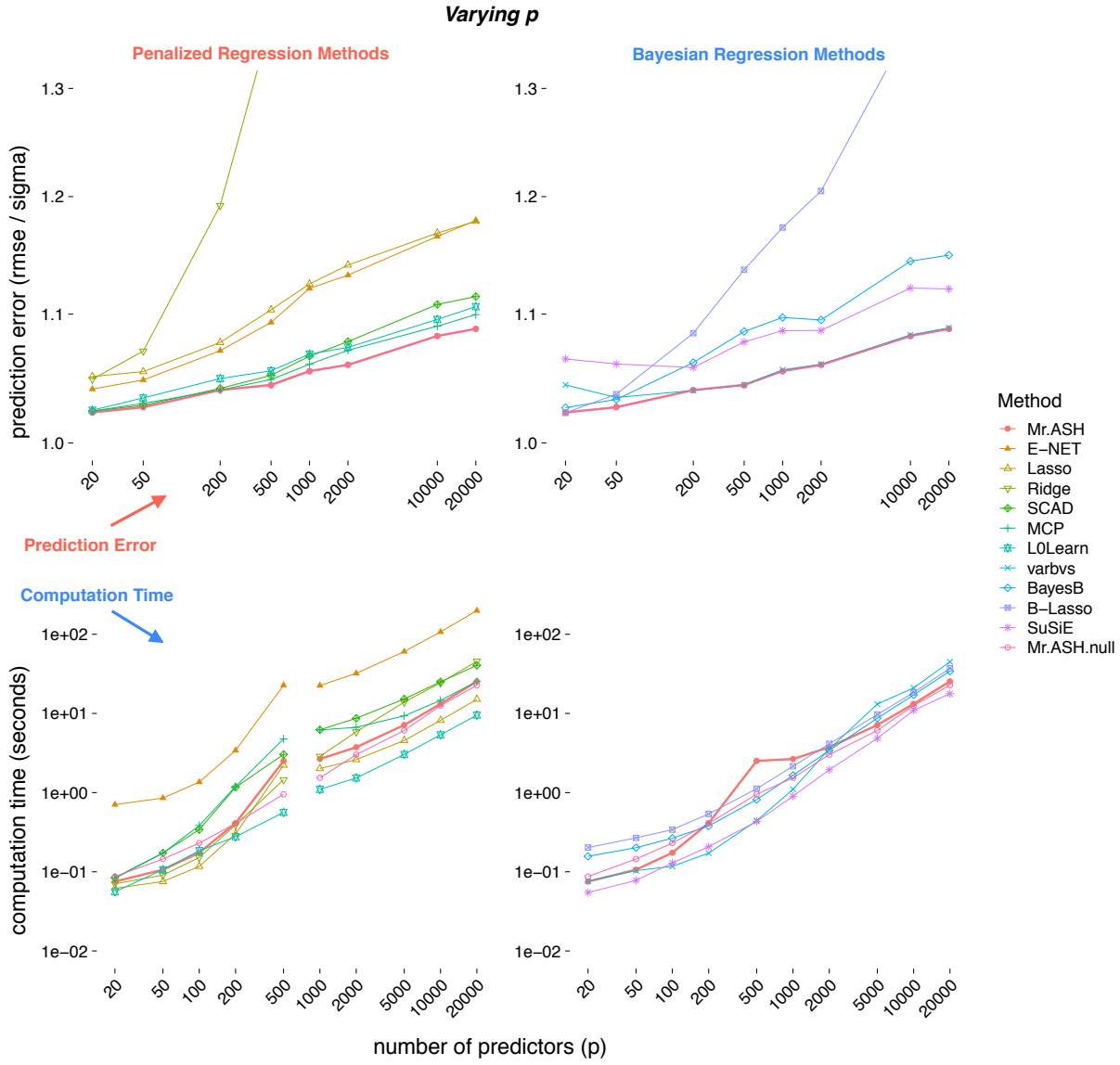


Figure 2.5: Comparisons of prediction error *(Top)* and computation time *(Bottom)* for both penalized regression methods *(Left)* and Bayesian regression methods *(Right)*. Simulations are performed by varying p from 20 to 20,000 in the *Baseline* scenario (hence $s = 20$). We disconnect the prediction curves in the bottom left panel because the computation time of some penalized linear regression methods behaves differently for $p \leq n$ and for $p > n$. All points are averages over 20 independent simulations. The compute time for Mr.ASH includes the compute time for Lasso, which is used as an initialization. Using a simple null initialization (Mr.ASH.null) is slightly faster Mr.ASH.null.

underlying Mr.ASH is sub-optimal in these settings, which exhibit very strong correlations (≥ 0.95) among predictors (in contrast B-Lasso uses MCMC and SuSiE uses a different variational approximation that is specifically adapted to correlated designs).

These results demonstrate the versatility of Mr.ASH, and the effectiveness of the VEB approach to adapt itself to the sparsity level by learning the prior (and hence penalty) from the data.

Performance with varying sample size (n), signal strength (PVE), and signal shape (h): The results for this group of simulations (Figure 2.4) show that relative prediction accuracy of different methods does not vary so greatly with n, PVE or h as it does with s (e.g. there are fewer lines crossing one another here than in results with varying s). The most notable exception is varying h in the dense setting ($p = 200$, $s = 200$), where for example Ridge and B-Lasso perform well for lighter-tailed signals, but poorly for heavier-tailed signals. Our explanation for this is that in this setting h determines the *effective sparsity*. For example, when $h = t_1$ the effect sizes are very long tailed, and so one or two coefficient may have much larger absolute magnitude than the others. Methods that are well adapted to sparse settings do well in this setting, but worse in settings where effects are more similar across predictors (e.g. $h =$ Constant).

Performance and computational scaling with varying dimension (p): For this group of simulations we show results for both prediction accuracy and computational time (Figure 2.5). The computation time results show that, with the software settings used here, all the methods scale similarly with increasing p. E-NET is substantially slower than the other methods because we chose to tune both tuning parameters of E-NET by cross-validation (CV), whereas for the other CV-based methods we tune only one tuning parameter. E-NET could be run faster by tuning only one parameter, but potentially at loss of prediction accuracy in some settings. Although Mr.ASH estimates its prior from a very flexible family – effectively tuning a large number of tuning parameters – it is approximately as fast as the methods that tune a single parameter by CV. This highlights a computational advantage of tuning parameters by EB rather than by CV.

We note that the Mr.ASH computation time here is dominated by the time spent computing the

Table 2.3: Average computation time for each method across all simulations (Groups 1-3). Note: the computation time of Mr.ASH includes the Lasso computation time for initialization; computation time with null initialization is shown as Mr.ASH.null.

Mr.ASH	Mr.ASH.null	E-NET	Lasso	Ridge	SCAD
20.37	15.05	223.66	18.79	33.76	35.04
MCP	L0Learn	varbvs	SuSiE	BayesB	B-Lasso
22.00	7.78	54.98	23.59	24.80	16.07

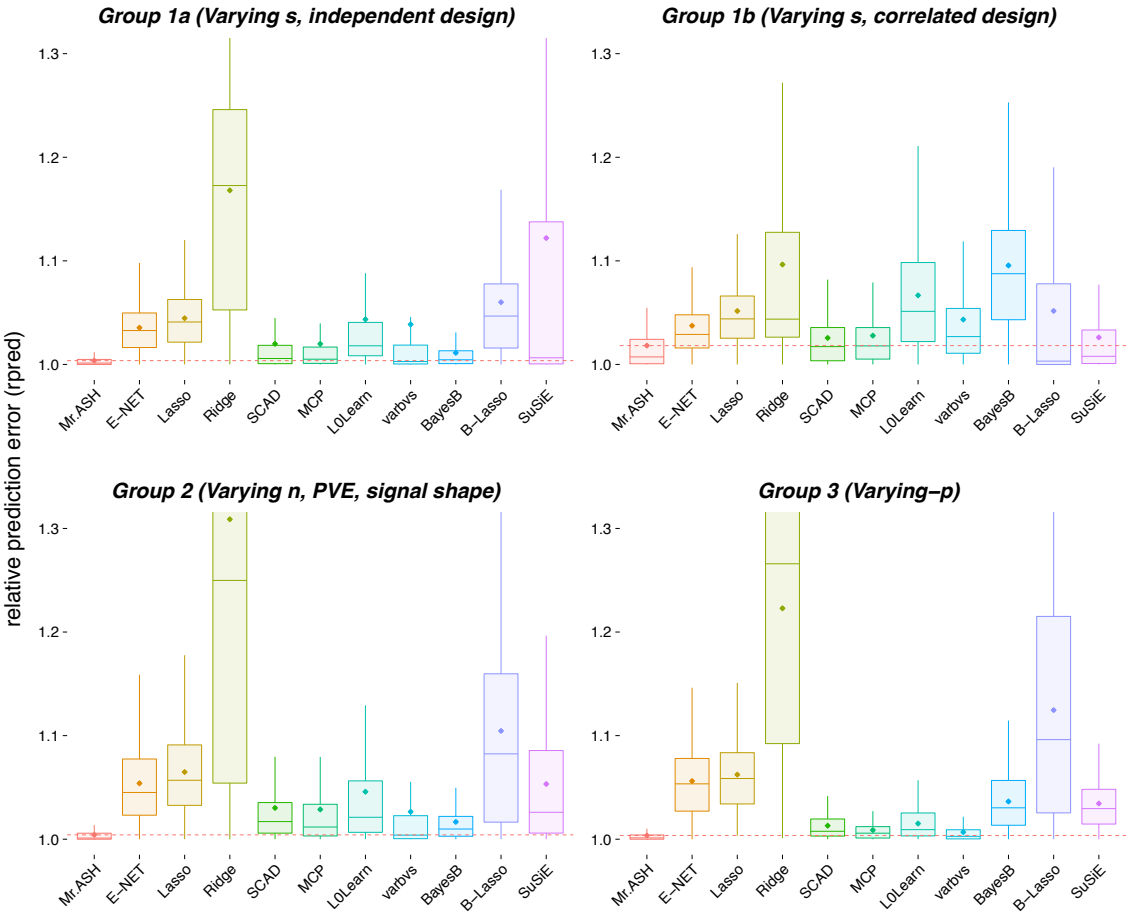


Table 2.4: Boxplots of RRMSEs (2.34), which measure prediction accuracy relative to the best performing method in each data set. Each panel shows results for a group of simulations. The horizontal line and dot in each box indicates median and mean performance respectively, and the limits of the box indicate the inter-quartile range.

Lasso solution, which we use for initialization. However, running Mr.ASH from the null initialization requires similar computation time compared to Lasso. This is because the Lasso initialization provides a "warm start" that not only helps Mr.ASH find a better solution, but also reduces the number of iterations required for convergence.

It may seems surprising that the computation of the MCMC-based methods (B-Lasso, BayesB) scales similarly to the other methods. However, this is simply because the software defaults for these methods increase the number of MCMC updates linearly with p (and each update has cost independent of p). Note that, as p increases, the prediction accuracy of these methods become substantially less competitive. For B−Lasso this may partly reflect the fact that increasing p corresponds to increasing sparsity (because s is fixed here), and B-Lasso is not well-adapted to sparse settings. However, for BayesB it seems likely that the poorer prediction performance for large p reflects a failure of the MCMC to converge. If so then this could be rectified by increasing computation time. (Still, it is interesting to note that BayesB performance is competitive here with Lasso for roughly the same computational effort.)

Overall Mr.ASH shows consistently the best prediction accuracy, with the computational time being competitive with other methods.

2.5.3 Overview of the Empirical Results

To provide a high level summary of the relative prediction accuracy of all methods across all simulations we normalize the performance in each data set to the best-performing method. That is, if RMSE_{mti} denotes the RMSE for method m under scenario t for data set (replicate) i, we define its relative prediction performance:

$$\text{RRMSE}_{mti} = \frac{\text{RMSE}_{mti}}{\min_m \{\text{RMSE}_{mti}\}} \tag{2.34}$$

Thus a method performs similarly to the best method if its RRMSE value is close to 1.

The results (Figure 2.4) emphasise the overall competitive performance of Mr.ASH compared

with all other methods. They also show that Mr.ASH performs well not only on average in most scenarios, but also consistently well across data sets in each scenario. Indeed, for Groups 1a, 3 and 4 its performance is very close to the best method for almost every data set (the box plots show very little deviation from 1). This highly competitive prediction performance is obtained with similar average computation time to the other methods (Table 2.3).

2.5.4 Impact of Initialization and Update Order

Since Mr.ASH is solving a non-convex optimization problem, the initialization of this coordinate ascent algorithm can affect the solution obtained, and hence predictive performance. Also, the order of the coordinate updates of $q_1, \cdots, q_p$ can also affect the solution obtained [123]. As shown in the previous section, initializing Mr.ASH at the (cross-validated) Lasso estimate of $\mathbf{b}$ produces consistently reliable prediction performance, and with only a minor increase in computation compared with Lasso. Here we illustrate the importance of this warm start in situations where $\mathbf{X}$ has highly correlated columns, and examine the impact of update order.

Specifically we assessed the following four different initialization and update order combinations:

- Mr.ASH.init : Lasso CV initialization, random update order. This is our default setting, presented as Mr.ASH above.
- Mr.ASH.null : null initialization ($\mathbf{b} = 0$), random update order.
- Mr.ASH.order : null initialization, Lasso path update order (i.e. the coordinates are updated in the order that they enter the Lasso path, from first to last).
- Mr.ASH.init&order : Lasso CV initialization, Lasso path update order.

Note that Mr.ASH.order reduces up-front computation compared with Mr.ASH.init by avoiding the cross-validation step in the Lasso.

The main factor that influences the need for warm start is the correlations among the predictors (columns of $\mathbf{X}$). Therefore we conducted simulations under the *Baseline* scenario, but with increasing

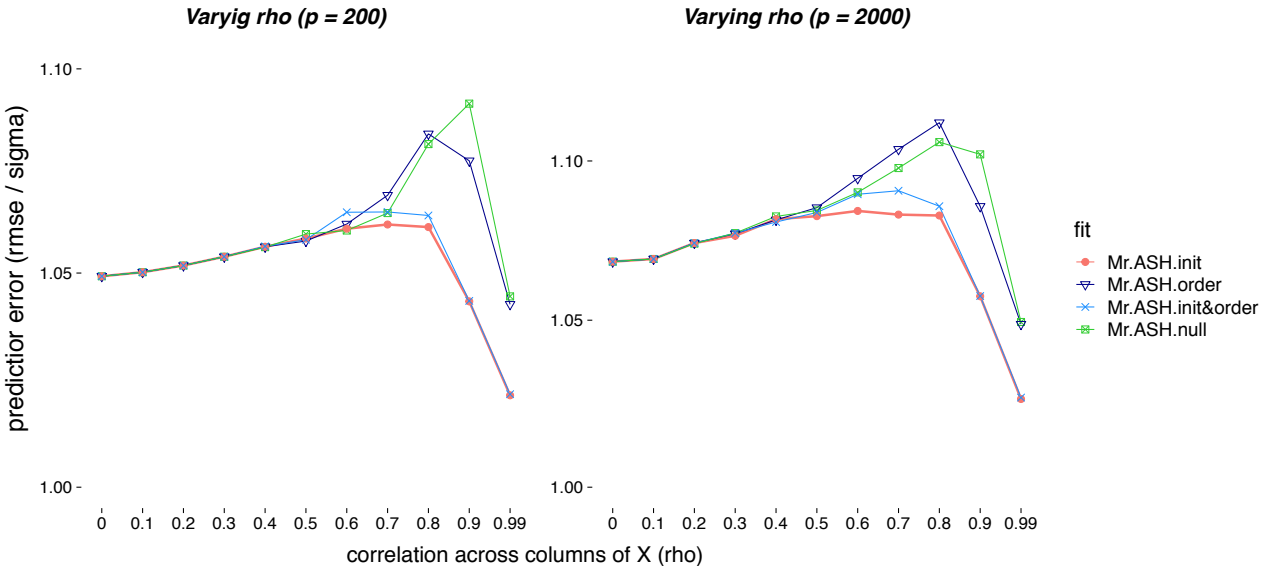


Figure 2.6: Prediction errors (2.33) (RMSE/σ) of Mr.ASH methods with different settings for initialization and update order.

correlations among the predictors (design set to Equicorr(ρ) for $\rho = 0, 0.1, \ldots, 0.9, 0.99$). The results (Figure 2.6) clearly demonstrates that using Lasso CV initialization can improve prediction accuracy, specifically for settings where $\rho > 0.6$. For smaller values of ρ all methods perform similarly, consistent with the expectation that the optimization objective function will typically be better behaved – closer to convex – in such settings [23]. In these simulations use of the Lasso path update order did not improve accuracy.

2.6 Conclusions

We have presented a new Variational Empirical Bayes method for multiple linear regression, particularly designed for fast and accurate prediction. This VEB method combines flexible shrinkage priors with variational methods for fast posterior approximation. Variational methods and Empirical Bayes methods are sometimes disparaged because of their tendency to under-state uncertainty compared with "fully Bayesian" methods ([110], [160] and references therein). However, in some applications uncertainty is of secondary importance compared with speed and accuracy of point estimates. For

example, this may occur when multiple regression is used simply to build an accurate predictor for downstream use [e.g. 50], and our VEB approach seems particularly attractive for such settings.

A natural next step would be to produce similar VEB methods for non-Gaussian (generalized) linear models [105]. Extension of our methods to logistic regression should be possible via additional variational approximations [77]. Extension to other types of outcome distribution (e.g. Poisson) may require more work.

More generally, our work here provides an example of the benefits of an EB approach in an important statistical problem. While EB approaches have been argued to be attractive both in theory [80] and in practice [41], ultimately they have not been widely used outside of the simple normal means model and wavelet shrinkage [82], and for moderating variance estimates in gene expression studies [138, 101]. Recent work has emphasized the potential for EB methods for several other applications, including: smoothing non-gaussian data [171], multiple testing [144, 148, 157, 57], matrix factorization and sparse factor analysis [164] and additive models [162]. We hope that these examples, including our work here, may help spur the wider implementation and adoption of convenient EB approaches for other problems.

2.7 Supplementary Material

2.7.1 Preliminaries: results for the normal means model

In the appendices we state and prove slightly more general results than those given in the main chapter. In particular we generalize the results to relax the assumption $\mathbf{x}_j^T \mathbf{x}_j = 1$. To do so we need results for a more general version of the normal means model than we considered in the main chapter. In this appendix we define this NM model and state the main results.

The normal means model

Let $\text{NM}_n(f, s_1^2, \cdots, s_n^2)$ denote the normal means model with prior f and observation-specific variances $\text{s}^2 = (s_1^2, \cdots, s_n^2) \in \mathbb{R}_+^n$. That is, the model

$$\begin{aligned} y_j \,|\, b_j, s_j^2 &\sim \mathcal{N}(\,\cdot\,; b_j, s_j^2), \quad j = 1, \cdots, n, \\ b_j \,|\, f &\overset{iid}{\sim} f(\,\cdot\,) \quad j = 1, \cdots, n, \end{aligned} \tag{2.35}$$

with $y_j, b_j \in \mathbb{R}$.

We will make use of mixture priors, $f \in \mathcal{G}(u_1^2, \ldots, u_K^2)$; that is,

$$\begin{aligned} f(\,\cdot\,) &= \sum_{k=1}^{K} \pi_k f_k(\,\cdot\,) \\ f_k(\,\cdot\,) &\triangleq \mathcal{N}(\,\cdot\,; 0, u_k^2). \end{aligned} \tag{2.36}$$

As in (2.37), it is convenient to introduce the latent variable representation:

$$\begin{aligned} b_j \,|\, f, \gamma_j = k &\sim f_k(\,\cdot\,) \\ \Pr(\gamma_j = k \,|\, f) &= \pi_k. \end{aligned} \tag{2.37}$$

We will write the joint prior for b_j and γ_j as

$$p_{\text{prior}}(b_j, \gamma_j = k) \triangleq p(b_j, \gamma_j = k \,|\, f) = \pi_k f_k(b_j). \tag{2.38}$$

We will also write $p_{\text{prior}}(f)$ for this distribution when we want to make explicit its dependence on f.

Posterior distributions under normal means model

We let $q^{\mathrm{NM}}(\,\cdot \,|\, y, f, s^2)$ denote the posterior distributions for b, γ under the model with a single observation $\mathrm{NM}_1(f, s^2)$:

$$y \,|\, b, s^2 \sim \mathcal{N}(\,\cdot\,; b, s^2), \quad b_j \,|\, f \overset{iid}{\sim} f(\,\cdot\,).$$

For $f \in \mathcal{G}(u_1^2, \ldots, u_K^2)$ these posteriors can be written as:

$$\begin{aligned} q^{\mathrm{NM}}(b, \gamma = k \,|\, y, f, s^2) &= \phi_k(y; f, s^2)\, \mathcal{N}\bigg(b\,; \mu_k(y; f, s^2), \frac{s^2 u_k^2}{s^2 + u_k^2}\bigg), \\ q^{\mathrm{NM}}(b \,|\, y, f, s^2) &= \sum_{k=1}^{K} q^{\mathrm{NM}}(b, \gamma = k \,|\, y, f, s^2), \end{aligned} \tag{2.39}$$

where

$$\mu_k(y; f, s^2) \triangleq u_k^2 y/(s^2 + u_k^2) \tag{2.40}$$

$$\ell_k(y; f, s^2) \triangleq p(y \,|\, f_k, s^2) = \int p(y \,|\, b, s^2) p(b \,|\, f_k) db \tag{2.41}$$

$$\phi_k(y; f, s^2) \triangleq \frac{\pi_k \ell_k(y; f, s^2)}{\sum_{l=1}^{K} \pi_l \ell_l(y; f, s^2)}. \tag{2.42}$$

are the operators for the mean, the component marginal log-likelihood, and the responsibility. We also define the NM posterior mean operator under the model $\mathrm{NM}_n(f, s_1^2, \cdots, s_n^2)$.

Definition 2.7.1 (Normal Means Posterior Mean Operator)**.** *We define the NM posterior mean operator, $\widetilde{S}_{f,s^2} : \mathbb{R} \to \mathbb{R}$, to be the mapping*

$$\widetilde{S}_{f,s^2}(y) \triangleq \mathbb{E}(b \,|\, y, f, s^2) \tag{2.43}$$

where $\mathbb{E}$ denotes the expectation under the model $\mathrm{NM}_1(f, s^2)$. Therefore, from (2.40) *and* (2.42)

we have:

$$\widetilde{S}_{f,s^2}(y) = \sum_{k=1}^{K} \phi_k(y; f, s^2)\mu_k(y; f, s^2).$$

Note that quantities defined in the main text, including (2.10), (2.11), and (2.13), can be obtained from these definitions by settings $f = g_\sigma$ and $s^2 = \sigma^2$. Similarly, $p^{\mathrm{NM}}_{\mathrm{post}}(b \,|\, y, g, \sigma^2)$ defined in (2.14) is a special case of q^{NM}:

$$p^{\mathrm{NM}}_{\mathrm{post}}(\,\cdot \,|\, y, g, \sigma^2) = q^{\mathrm{NM}}(\,\cdot \,|\, y, g_\sigma, \sigma^2). \tag{2.44}$$

Also, $S_{g,\sigma}$ in Definition 2.4.1 is equivalent to $\widetilde{S}_{g_\sigma,\sigma^2}$ in Definition 2.7.1.

Evidence Lower Bound

Let F^{NM} denote the Evidence Lower Bound (ELBO) [e.g. 16] for $\mathrm{NM}_1(f, s^2)$:

$$\begin{aligned} F^{\mathrm{NM}}(q, f, s^2) &= \log p(y \,|\, f, s^2) - D_{KL}(q \,\|\, q^{\mathbf{NM}}) \\ &= \mathbb{E}_q \log p(y \,|\, b, s^2) - D_{KL}(q \,\|\, p_{\mathrm{prior}}(f)) \\ &= -\frac{1}{2s^2}\mathbb{E}_q(y-b)^2 - D_{KL}(q \,\|\, p_{\mathrm{prior}}(f)). \end{aligned} \tag{2.45}$$

Here q is an arbitrary probability density for $b \in \mathbb{R}$ and $\gamma \in \{1, \cdots, K\}$, and recall q^{NM} denotes the true posterior. From the equality condition of the Jensen's inequality [83], F^{NM} is optimized over q by the true posterior q^{NM}, which leads to the following lemma:

Lemma 2.7.1 (NM Posterior as maximizer of ELBO)**.** *The posterior q^{NM} solves the following optimization problem:*

$$q^{\mathrm{NM}}(\,\cdot \,|\, y, f, s^2) = \operatorname*{argmax}_q -\frac{1}{2s^2}\mathbb{E}_q(y-b)^2 - D_{KL}(q \,\|\, p_{\mathrm{prior}}(f)).$$

Now, we consider the ELBO for the normal means model with n observations, $\mathrm{NM}_n(f, \mathbf{s}^2 = (s_1^2, \cdots, s_n^2))$. For $q \in \mathcal{Q}$ ($q = (q_1, \ldots, q_n)$ say), the ELBO is simply the sum of the individual

ELBOs:

$$F^{\mathrm{NM}}(q, f, \mathbf{s}^2) = \sum_{j=1}^{p} F^{\mathrm{NM}}(q_j, f, s_j^2). \tag{2.46}$$

Consequently, the optimal q_j is, by Lemma 2.7.1, $q_j(b, \gamma) = q^{\mathrm{NM}}(b, \gamma \,|\, y_j, f, s_j^2)$, and so from (2.39) the optimal q_j is of the parametric form:

$$q_j(b, \gamma = k) = \phi_{jk}\, \mathcal{N}(b; m_{jk}, v_{jk}), \quad j = 1, \cdots, p. \tag{2.47}$$

For any $q = (q_1, \ldots, q_n)$ where q_j has the parametric form (2.47) the ELBO has an analytic form, Specifically, exploiting the analytic expression of the Kullback-Leibler divergence between two normal distributions [e.g. 72], we have

$$F^{\mathrm{NM}}(\boldsymbol{\phi}, \mathbf{m}, \mathbf{v}, \boldsymbol{\pi}, \sigma^2) = \sum_{j=1}^{p} F_j^{\mathrm{NM}}(\boldsymbol{\phi}_j, \mathbf{m}_j, \mathbf{v}_j, \boldsymbol{\pi}, \sigma^2)$$

with

$$F_j^{\mathrm{NM}}(\boldsymbol{\phi}_j, \mathbf{m}_j, \mathbf{v}_j, \boldsymbol{\pi}, \sigma^2) \tag{2.48}$$

$$= -\frac{1}{2}\log(2\pi s_j^2) - \frac{1}{2s_j^2}(y - \bar{b}_j)^2 - \frac{1}{2s_j^2}\left[\sum_{k=1}^{K} \phi_{jk}\left(m_{jk}^2 + v_{jk}\right) - \bar{b}_j^2\right] \tag{2.49}$$

$$- \sum_{k=1}^{K} \phi_{jk} \log\frac{\phi_{jk}}{\pi_k} + \frac{1}{2}\sum_{k\,:\,u_k^2>0} \phi_{jk}\left[1 + \log\frac{v_{jk}}{u_k^2} - \frac{m_{jk}^2 + v_{jk}}{u_k^2}\right], \tag{2.50}$$

with $\bar{b}_j = \sum_k \phi_{jk} m_{jk}$. The parametric expression in the first line is for the expected log-likelihood $\mathbb{E}_{q_j} \log p(y_j \,|\, b_j, \sigma^2)$, and the expression in the second line is for the negative Kullback-Leibler divergence term $-D_{KL}(q_j \,\|\, p_{\mathrm{prior}}(f))$.

2.7.2 Derivation of Algorithm 2 (Proof of Proposition 2.3.1)

We now state and prove Proposition 1′, a more general version of Proposition 2.3.1 that relaxes the assumption $\mathbf{x}_j^T \mathbf{x}_j = 1$. For notational simplicity, we let $d_j = \mathbf{x}_j^T \mathbf{x}_j$ for $j = 1, \cdots, p$.

Proposition 1′. *Let q_{-j} denote all elements of q except q_j. Let $\bar{\mathbf{r}}_j$ denote the expectation, under q_{-j}, of the residuals computed using all variables except j:*

$$\bar{\mathbf{r}}_j \triangleq \mathbf{y} - \mathbf{X}_{-j} \bar{\mathbf{b}}_{-j} \in \mathbb{R}^n, \tag{2.51}$$

where $\mathbf{X}_{-j}$ denote the matrix $\mathbf{X}$ excluding the jth column, and $\bar{\mathbf{b}}_{-j}$ denotes the expectation (i.e. first moment) of q_{-j}. Let $\widetilde{b}_j$ denote the univariate ordinary least squares (OLS) estimator of b_j when regressing $\bar{\mathbf{r}}_j$ against $\mathbf{x}_j$:

$$\widetilde{b}_j \triangleq \frac{\mathbf{x}_j^T \bar{\mathbf{r}}_j}{d_j} \in \mathbb{R} \tag{2.52}$$

where $d_j \triangleq \mathbf{x}_j^T \mathbf{x}_j$. Then

Part 1. The optimum $\widehat{q}_j \triangleq \operatorname{argmax}_{q_j} F(q_j, q_{-j}, g, \sigma^2)$ is achieved by $\widehat{q}_j = q^{\mathrm{NM}}(\,\cdot\,|\widetilde{b}_j, g_\sigma, \sigma^2/d_j)$ defined in (2.39)*, which is the posterior distribution of b under the following NM model:*

$$\widetilde{b}_j \,|\, b, \sigma^2 \sim \mathcal{N}(\,\cdot\,; b, \sigma^2/d_j), \quad b \,|\, g, \sigma^2 \sim g_\sigma(\,\cdot\,). \tag{2.53}$$

Part 2. The optimum $\widehat{g} \triangleq \operatorname{argmax}_{g \in \mathcal{G}(\sigma_1^2, \ldots, \sigma_K^2)} F(q, g, \sigma^2)$ is achieved by

$$\widehat{g} = \sum_{k=1}^{K} \widehat{\pi}_k g_k, \quad \textit{where} \quad \widehat{\pi}_k = \frac{1}{p} \sum_{j=1}^{n} q_j(\gamma_j = k) \;\; \textit{for} \;\; k = 1, \cdots, K. \tag{2.54}$$

In particular, if each variational factor q_j is updated as Part 1 above, meaning that $q_j(\,\cdot\,) = q^{\mathrm{NM}}(\,\cdot\,|\widetilde{b}_j, g_\sigma, \sigma^2/d_j)$ for $j = 1, \cdots, p$, then each $q_j(\gamma_j = k)$ is equal to the responsibility $\phi_k(\widetilde{b}_j; g_\sigma, \sigma^2/d_j)$ (2.42)*.*

Part 3. The optimum $\widehat{\sigma}^2 \triangleq \operatorname{argmax}_{\sigma^2 \in \mathbb{R}_+} F(q, g, \sigma^2)$, under the parametrization (2.47) *of q, is*

achieved by:

$$\sigma^2 \leftarrow \frac{\Big[\|\mathbf{y}-\mathbf{X}\bar{\mathbf{b}}\|^2 + \sum_{j=1}^p \sum_{k\,:\,\sigma_k^2>0} \phi_{jk}\big(d_j + (1/\sigma_k^2)\big)(m_{jk}^2 + v_{jk}) - \sum_{j=1}^p d_j \bar{b}_j^2\Big]}{n + p(1-\pi_1)}$$

In particular, if $q_j(\,\cdot\,) = q^{\mathrm{NM}}(\,\cdot\,|\widetilde{b}_j, g_\sigma, \sigma^2/d_j)$*, we have a simpler update rule:*

$$\widehat{\sigma}^2 = \frac{1}{n + p(1-\pi_1)}\bigg[\|\mathbf{y}-\mathbf{X}\bar{\mathbf{b}}\|^2 + \sum_{j=1}^p d_j(\widetilde{b}_j - \bar{b}_j)\bar{b}_j + p(1-\pi_1)\sigma^2\bigg], \tag{2.55}$$

where each $\bar{b}_j$ *is the expectation of* q_j*, i.e.* $\bar{b}_j = \widetilde{S}_{g_\sigma,\sigma^2/d_j}(\widetilde{b}_j)$ *(see Definition 2.7.1.).*

Proposition 1′ parses the coordinate ascent algorithm, which maximizes the ELBO:

$$\begin{aligned} F(q,g,\sigma^2) &= \mathbb{E}_q \log p(\mathbf{y}\,|\,\mathbf{X},g,\sigma^2) - D_{KL}(q\,\|\,p_{\mathrm{post}}) \\ &= \mathbb{E}_q \log p(\mathbf{y}\,|\,\mathbf{X},\mathbf{b},\sigma^2) - \sum_{j=1}^p D_{KL}(q_j\,\|\,p_{\mathrm{prior}}(g_\sigma)) \\ &= -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\mathbb{E}_q\,\|\mathbf{y}-\mathbf{X}\mathbf{b}\|^2 - \sum_{j=1}^p D_{KL}(q_j\,\|\,p_{\mathrm{prior}}(g_\sigma)). \end{aligned} \tag{2.56}$$

This follows from the Bayes rule

$$p_{\mathrm{post}}(\mathbf{b},\boldsymbol{\gamma}) = \frac{p(\mathbf{y}\,|\,\mathbf{X},\mathbf{b},\sigma^2)p(\mathbf{b},\boldsymbol{\gamma}\,|\,g,\sigma^2)}{p(\mathbf{y}\,|\,\mathbf{X},g,\sigma^2)}$$

and that $q(\mathbf{b},\boldsymbol{\gamma}) = \prod_{j=1}^p q_j(b_j,\gamma_j)$ and $p(\mathbf{b},\boldsymbol{\gamma}\,|\,g,\sigma^2) = \prod_{j=1}^p p_{\mathrm{prior}}(b_j,\gamma_j)$ factorize.

Each part of the Proposition 1′ is for the update of $q = (q_1,\cdots,q_p)$, g and σ^2 whose proof will be provided in Appendix 2.7.2, 2.7.2 and 2.7.2, respectively.

Update of q

In each inner loop iteration, we update q_j one at a time while the rest is fixed. That is,

$$q_j \leftarrow \operatorname*{argmax}_{q_j} F(q_j; q_{-j}, g, \sigma^2). \tag{2.57}$$

In what follows, we will illustrate an high-level idea of how the variational factor q_j will be updated. In practice, this abstract update should be concretized in a tangible form, such as a closed-form update formula for each variational parameter of q_j, which has not been specified yet.

Once we extract the terms relevant to q_j and remove the terms independent of q_j from the ELBO $F(q, g, \sigma^2)$, the inner loop update can be simplied using the univariate OLS estimator $\widetilde{b}_j$ (2.52) as

$$\begin{aligned} q_j \leftarrow q_j^{\mathrm{NM}} &= \operatorname*{argmax}_{q_j} \mathbb{E}_{q_j}\mathbb{E}_{q_{-j}} \log p(\mathbf{y} \mid \mathbf{X}, b_j, \mathbf{b}_{-j}, \sigma^2) - D_{KL}(q_j \,\|\, p_{\mathrm{prior}}) \\ &= \operatorname*{argmax}_{q_j} -\frac{d_j}{2\sigma^2}\mathbb{E}_{q_j}(b_j - \widetilde{b}_j)^2 - D_{KL}(q_j \,\|\, p_{\mathrm{prior}}). \end{aligned} \tag{2.58}$$

Note that q_j^{NM} only depends on the data $(\mathbf{y}, \mathbf{X})$ through $\widetilde{b}_j$ (and d_j, which is known and fixed). Also, we note, in passing, that $\widetilde{b}_j = \overline{b}_j + (\mathbf{x}_j\overline{\mathbf{r}}/d_j)$ can be also understood as an expected gradient step.

Now, Lemma 2.7.1 proves the first part of Proposition 1′. Note that the right hand side of (2.58) is the ELBO for the model (2.53). Therefore,

$$q_j^{\mathrm{NM}}(\,\cdot\,) = q^{\mathrm{NM}}(\,\cdot \mid \widetilde{b}_j, g_\sigma, \sigma^2/d_j) = \sum_{k=1}^{K} \phi_{jk}\mathcal{N}\bigg(\,\cdot\,; \frac{d_j\widetilde{b}_j}{d_j + (1/\sigma_k^2)}, \frac{\sigma^2}{d_j + (1/\sigma_k^2)}\bigg), \tag{2.59}$$

and, in particular, the posterior mean $\overline{b}_j$ is updated by

$$\overline{b}_j = \sum_{k=1}^{K} \frac{\phi_{jk}d_j}{d_j + (1/\sigma_k^2)}\widetilde{b}_j.$$

Update of g

The prior g will be updated by

$$g \leftarrow \operatorname*{argmax}_{g \in \mathcal{G}} F(q, g, \sigma^2).$$

Let us recall that F is the sum of the expected likelihood $\mathbb{E}_q \log p(\mathbf{y} \mid \mathbf{X}, \mathbf{b}, \sigma^2)$ and the negative KL divergence $\sum_{j=1}^p D_{KL}(q_j \,\|\, p_{\text{prior}})$. The only term involving the update of g is the KL divergence term, which can be written in parametric form using (2.47) as

$$\sum_{j=1}^p D_{KL}(q_j \,\|\, p_{\text{prior}}) = \sum_{j,k} \phi_{jk} \log \frac{\pi_k}{\phi_{jk}} + \text{constant},$$

where the above constant term does not depend on $\boldsymbol{\pi}$, and $\phi_{jk} \triangleq q_j(\gamma_j = k)$. Therefore, we have

$$\pi_k \leftarrow \operatorname*{argmax}_{\pi_k} - \sum_j \phi_{jk} \log \pi_k = \frac{1}{p} \sum_{j=1}^p \phi_{jk}$$

$$g \leftarrow \sum_{k=1}^K \pi_k g_k.$$

If $q_j = q(\cdot \,|\, \widetilde{b}_j, g_\sigma, \sigma^2/d_j)$, then ϕ_{jk} is equal to the responsibility $\phi_k(\widetilde{b}_j; g_\sigma, \sigma^2/d_j)$, which is defined in (2.42). Note that this is equal to the one-step EM update for fitting the NM model (2.35) by EB [144], as discussed in Section 2.2.2. (Also, this is a typical Gaussian mixture EM step; see Bishop [16], Hastie et al. [72]) This completes the proof for the second part of Proposition $1'$.

Update of σ^2

The noise variance σ^2 be updated by

$$\sigma^2 \leftarrow \operatorname*{argmax}_{\sigma^2 \in \mathbb{R}_+} F(q, g, \sigma^2).$$

Note that the ELBO (2.56) can be written, using the optimal form (2.47), as:

$$F(\boldsymbol{\phi}, \mathbf{m}, \mathbf{v}, \boldsymbol{\pi}, \sigma^2) \tag{2.60}$$

$$= -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left\|\mathbf{y} - \mathbf{X}\bar{\mathbf{b}}\right\|^2 - \frac{1}{2\sigma^2}\sum_{j=1}^{p} d_j \left[\sum_k \phi_{jk}\left(m_{jk}^2 + v_{jk}\right) - \bar{b}_j^2\right] \tag{2.61}$$

$$-\sum_{j=1}^{p}\sum_{k=1}^{K}\phi_{jk}\log\frac{\phi_{jk}}{\pi_k} + \frac{1}{2}\sum_{j=1}^{p}\sum_{k>1}\phi_{jk}\left[1 + \log\frac{v_{jk}}{\sigma^2\sigma_k^2} - \frac{v_{jk} + m_{jk}^2}{\sigma^2\sigma_k^2}\right], \tag{2.62}$$

with $\bar{b}_j = \sum_k \phi_{jk} m_{jk}$. The parametric expression (2.61) in the second line is for the expected log-likelihood $\mathbb{E}_q \log p(\mathbf{y} \mid \mathbf{X}, \mathbf{b}, \sigma^2)$, and the expression (2.62) in the third line is for the negative Kullback-Leibler divergence term $-\sum_{j=1}^{p} D_{KL}(q_j \,\|\, p_{\text{prior}})$.

Then, the coordinate update formula for σ^2 is given by taking the partial derivative $\partial F/\partial\sigma^2$ of F with respect to σ^2 and find a unique zero of $\partial F/\partial\sigma^2$:

$$\sigma^2 \leftarrow \widehat{\sigma}^2 \text{ such that } \frac{\partial}{\partial\sigma^2}F(\boldsymbol{\phi}, \mathbf{m}, \mathbf{v}, \boldsymbol{\pi}, \sigma^2)\Big|_{\sigma^2=\widehat{\sigma}^2} = 0.$$

A simple calculation shows that

$$\sigma^2 \leftarrow \frac{1}{n + p(1-\pi_1)}\left[\|\mathbf{y} - \mathbf{X}\bar{\mathbf{b}}\|^2 + \sum_{j=1}^{p}\sum_{k>1}\phi_{jk}\big(d_j + (1/\sigma_k^2)\big)(m_{jk}^2 + v_{jk}) - \sum_{j=1}^{p} d_j\bar{b}_j^2\right] \tag{2.63}$$

Note that we can simpify (2.63) using the update (2.59) for q_j's:

$$v_{jk} = \frac{\sigma_{\text{old}}^2}{d_j + (1/\sigma_k^2)}, \quad m_{jk} = \frac{d_j\widetilde{b}_j}{d_j + (1/\sigma_k^2)},$$

which yields

$$\begin{aligned}\sum_{j=1}^{p}\sum_{k>1}\phi_{jk}\big(d_j + (1/\sigma_k^2)\big)(m_{jk}^2 + v_{jk}) &= \sum_{j=1}^{p}\sum_{k=1}^{K}\phi_{jk}m_{jk}d_j\widetilde{b}_j + \sum_{j=1}^{p}\sum_{k>1}^{K}\phi_{jk}\sigma_{\text{old}}^2 \\ &= d_j\bar{b}_j\widetilde{b}_j + p(1-\pi_1)\sigma_{\text{old}}^2.\end{aligned}$$

Therefore, we have:

$$\widehat{\sigma}^2 = \frac{1}{n + p(1-\pi_1)} \left[\|\mathbf{y} - \mathbf{X}\bar{\mathbf{b}}\|^2 + \sum_{j=1}^{p} d_j(\widetilde{b}_j - \bar{b}_j)\bar{b}_j + p(1-\pi_1)\sigma^2_{\text{old}} \right],$$

Here σ^2_{old} highlights that σ^2 is evaluated at the previous iterate in the right hand side. This proves the third part of Proposition $1'$.

Actual Implementation of the Algorithm

We summarize the actual implementation of the algorithm in Algorithm 4 when the assumptions $\mathbf{x}_j^T \mathbf{x}_j = 1$ for $j = 1, \cdots, p$ are removed.

2.7.3 VEB as a PLR

Normal means as a penalized problem

In this subsection we draw a connection between the normal means problem and a penalized approach. Specifically we write the posterior mean as a solution to a penalized least squares problem.

Consider the normal means model (2.35), i.e.

$$y_j \mid b_j, s_j^2 \sim \mathcal{N}(\cdot\,; b_j, s_j^2), \quad b_j \mid f \overset{iid}{\sim} f(\cdot), \quad j = 1, \cdots, n \tag{2.64}$$

and let F^{NM} denote the ELBO for this normal means model, i.e.

$$F^{\text{NM}}(q, f, s^2; y) = \mathbb{E}_q \log p(y \mid b, f, s^2) - D_{KL}(q \,\|\, p_{\text{prior}}(f)) \tag{2.65}$$

where $\mathbb{E}_q$ denotes the expectation over $b \sim q$. To lighten the notation, we will omit the dependency on y, and simply write $F^{\text{NM}}(q, f, s^2)$.

Algorithm 4 Actual Implementation of Algorithm 2 for $\mathcal{G}(\sigma_1^2, \cdots, \sigma_K^2)$.

Initialize: $\bar{\mathbf{b}}^{(0)}, \boldsymbol{\pi}^{(0)}, (\sigma^2)^{(0)}$.

$\bar{\mathbf{r}}^{(0)} \leftarrow \mathbf{y} - \mathbf{X}\bar{\mathbf{b}}^{(0)}$ (the expected residual).

0. Pre-computation

$$d_j = \mathbf{x}_j^T \mathbf{x}_j = \|\mathbf{x}_j\|^2, \quad \text{for} \quad j = 1, \cdots, p.$$

repeat

$\bar{\mathbf{r}}^{(t+1)} = \bar{\mathbf{r}}^{(t)}$, $\boldsymbol{\pi}^{(t+1)} = 0$.

for $j = 1, \cdots, p$ **do**

1. The expected gradient step.

$$\widetilde{b}_j \leftarrow \bar{b}_j^{(t)} + d_j^{-1} x_j^T r$$

2. The computation of ϕ_{jk}'s and the update of $\boldsymbol{\pi}$.

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

$$\phi_{jk} \leftarrow \frac{\pi_k p(\widetilde{b}_j \,|\, \gamma_j = k, (\sigma^2)^{(t)})}{\sum_k \pi_k p(\widetilde{b}_j \,|\, \gamma_j = k, (\sigma^2)^{(t)})} \propto \frac{\pi_k^{(t)}}{1 + d_j \sigma_k^2} \exp\left(\frac{d_j^2 \widetilde{b}_j^2}{2(\sigma^2)^{(t)}(d_j + (1/\sigma_k^2))} \right)$$

$$\pi_k^{(t+1)} \leftarrow \pi_k^{(t+1)} + p^{-1}\phi_{jk}$$

end for

3. The shrinkage step.

$$\bar{b}_j^{(t+1)} \leftarrow \sum_{k=1}^{K} \frac{\phi_{jk} d_j}{d_j + (1/\sigma_k^2)} \widetilde{b}_j,$$

4. The update of the expected residual.

$$\bar{\mathbf{r}}^{(t+1)} \leftarrow \bar{\mathbf{r}}^{(t+1)} + \mathbf{x}_j(\bar{b}_j^{(t)} - \bar{b}_j^{(t+1)})$$

end for

5. The update of σ^2.

$$(\sigma^2)^{(t+1)} \leftarrow \frac{1}{n + p(1 - \pi_1)} \left[\|\bar{\mathbf{r}}^{(t+1)}\|^2 + \sum_{j=1}^{p} d_j(\widetilde{b}_j - \bar{b}_j^{(t+1)})\bar{b}_j^{(t+1)} + p(1 - \pi_1)(\sigma^2)^{(t)} \right]$$

6. Delete $\boldsymbol{\phi}$ from memory and proceed to the next iteration: $t \leftarrow t + 1$.

until convergence criteria met

Output: $\bar{\mathbf{b}}^{(t)}, \boldsymbol{\pi}^{(t)}, (\sigma^2)^{(t)}$

With slight abuse of notation, we also define a function

$$F^{\mathrm{NM}}(\bar{b}, f, s^2) \triangleq \max_{q\,:\,\mathbb{E}_q(b)=\bar{b}} F^{\mathrm{NM}}(q, f, s^2), \tag{2.66}$$

where $\bar{b} \in \mathbb{R}$ a free variable. Here the symbol F^{NM} is overloaded, with its meaning being determined by its first argument ($\bar{b}$ or q), and the $\mathbb{E}_q(b) = \bar{b}$ means that the expectation (i.e. the mean or the first moment) of q is constrained to be $\bar{b}$. Thus, $F^{\mathrm{NM}}(\bar{b}, f, s^2)$ is an ELBO value evaluated at an "optimal" q having the expectation $\bar{b}$.

We remark that $F^{\mathrm{NM}}(\bar{b}, f, s^2)$ attains its global maximum at $\bar{b} = \widetilde{S}_{f,s^2}(y)$, and $F(q, f, s^2)$ attains its global maximum at $q = p^{\mathrm{NM}}_{\mathrm{post}}$. The relationship between them is

$$\begin{aligned} F(p^{\mathrm{NM}}_{\mathrm{post}}, f, s^2) &= \max_q F^{\mathrm{NM}}(q, f, s^2) = \max_{\bar{b}\in\mathbb{R}} \max_{q\,:\,\mathbb{E}_q(b)=\bar{b}} F^{\mathrm{NM}}(q, f, s^2) \\ &= \max_{\bar{b}\in\mathbb{R}} F^{\mathrm{NM}}(\bar{b}, f, s^2) = F^{\mathrm{NM}}(\widetilde{S}_{f,s^2}(y), f, s^2). \end{aligned} \tag{2.67}$$

Now, the following Lemma shows that $-F^{\mathrm{NM}}(\bar{b}, f, s^2)$ is the sum of the L_2-loss and the explicitly constructed penalty.

Lemma 2.7.2. *The ELBO $F^{\mathrm{NM}}(\bar{b}, f, s^2)$ can be written as a penalized log-likelihood:*

$$F^{\mathrm{NM}}(\bar{b}, g, \sigma^2) = -\frac{1}{2}\log(2\pi s^2) - \frac{1}{2s^2}(y-\bar{b})^2 - \widetilde{\rho}_{f,s^2}(\bar{b}) \tag{2.68}$$

where the penalty term is

$$\widetilde{\rho}_{f,s^2}(\bar{b}) \triangleq \min_{q\,:\,\mathbb{E}_q(b)=\bar{b}} \left[\frac{1}{2s^2}\operatorname{Var}(q) + D_{KL}(q \,\|\, p_{\mathrm{prior}}(f))\right]. \tag{2.69}$$

Further, for any $y \in \mathbb{R}$ this penalty term satisfies

$$(\widetilde{\rho}_{f,s^2})'(\widetilde{S}_{f,s^2}(y)) = (1/s^2)\left(y - \widetilde{S}_{f,s^2}(y)\right). \tag{2.70}$$

Assuming the inverse of $\widetilde{S}_{f,s^2}$ exists this implies that:

$$(\widetilde{\rho}_{f,s^2})'(b) = (1/s^2)\left(\widetilde{S}^{-1}_{f,s}(b) - b\right). \tag{2.71}$$

Proof. From (2.65) we have:

$$\begin{aligned} & F^{\mathrm{NM}}(q, f, s^2) \\ = & -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2s^2}\mathbb{E}_q(y-b)^2 - D_{KL}(q \,\|\, p_{\mathrm{prior}}(f)) \\ = & -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2s^2}\big(y - \mathbb{E}_q(b)\big)^2 - \left[\frac{1}{2s^2}\,\mathrm{Var}(q) + D_{KL}(q \,\|\, p_{\mathrm{prior}}(f))\right] \end{aligned}$$

The expressions (2.68, 2.69) then follow directly from the definition (2.66) of $F^{\mathrm{NM}}(\bar{b}, f, s^2)$. Expression (2.70) is a direct consequence of the fact that $F^{\mathrm{NM}}(\bar{b}, f, s^2)$ is optimized at $\bar{b} = \widetilde{S}_{f,s^2}(y)$ for any $y \in \mathbb{R}$, i.e.

$$\widetilde{S}_{f,s^2}(y) = \operatorname*{argmax}_{\bar{b}\in\mathbb{R}} F^{\mathrm{NM}}(\bar{b}, f, s^2),$$

hence the partial derivative of (2.68) with respect to $\bar{b}$ evaluated at $\bar{b} = \widetilde{S}_{f,s^2}(y)$ must be 0. Finally, expression (2.71) comes from setting $y = \widetilde{S}^{-1}_{f,s^2}(b)$ in (2.70). □

VEB as a penalized regression problem

Next, we consider the ELBO F (2.19) for multiple linear regression and define:

$$F(\bar{\mathbf{b}}, g, \sigma^2) \triangleq \max_{q\,:\,\mathbb{E}_q(\mathbf{b})=\bar{\mathbf{b}}} F(q, g, \sigma^2). \tag{2.72}$$

Again, the symbol F is overloaded, meaning that $F(\bar{\mathbf{b}}, g, \sigma^2)$ is an ELBO $F(q, g, \sigma^2)$ value evaluated at an optimal q such that $\mathbb{E}_q(\mathbf{b}) = \bar{\mathbf{b}}$. Also, $\bar{\mathbf{b}} \in \mathbb{R}^p$ is a free parameter.

The following Proposition is a direct consequence of (2.56) and Lemma 2.7.2.

Proposition 2.7.1. *The ELBO* (2.19) *can be written as a penalized log-likelihood:*

$$F(\bar{\mathbf{b}}, g, \sigma^2) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{X}\bar{\mathbf{b}}\|^2 - \sum_{j=1}^{p} \widetilde{\rho}_{g_\sigma, \sigma^2/d_j}(\bar{b}_j), \tag{2.73}$$

where the penalty function $\widetilde{\rho}$ is defined as in Lemma 2.7.2.

Proof. From (2.56), we have:

$$\begin{aligned} F(q, g, \sigma^2) = & -\frac{n-p}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{X}\mathbb{E}_q(\mathbf{b})\|^2 + \frac{1}{2\sigma^2}\sum_{j=1}^{p} d_j(\widetilde{b}_j - \mathbb{E}_{q_j}(b_j))^2 \\ & + \sum_{j=1}^{p}\left[-\frac{1}{2}\log(2\pi\sigma^2) - \frac{d_j}{2\sigma^2}\mathbb{E}_{q_j}(b_j - \widetilde{b}_j)^2 - D_{KL}(q_j \,\|\, p_{\text{prior}}(g_\sigma))\right] \\ & = -\frac{n-p}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{X}\mathbb{E}_q(\mathbf{b})\|^2 + \frac{1}{2\sigma^2}\sum_{j=1}^{p} d_j(\widetilde{b}_j - \mathbb{E}_{q_j}(b_j))^2 \\ & + \sum_{j=1}^{p} F^{\text{NM}}(q_j, g_\sigma, \sigma^2/d_j; \widetilde{b}_j) \end{aligned}$$

Threfore, the result directly follows from Lemma 2.7.2. □

2.7.4 Proofs

Proof. **of Proposition 2.3.2**

By Proposition 2.7.1 of Bertsekas [12], the sequence of iterates $\{q^{(t)}, g^{(t)}, (\sigma^2)^{(t)}\}$ for $t = 1, 2, \cdots$ from the coordinate updates converges monotonically to a local maximum of F, provided that F is continuously differentiable and each coordinate update

$$\begin{aligned} q_j^{(t+1)} &= \underset{q_j}{\text{argmax}}\ F(q_1^{(t+1)}, \cdots, q_{j-1}^{(t+1)}, q_j, q_{j+1}^{(t)}, \cdots, q_p^{(t)}, g^{(t)}, (\sigma^2)^{(t)}) \\ g^{(t+1)} &= \underset{g \in \mathcal{G}}{\text{argmax}}\, F(q^{(t+1)}, g, (\sigma^2)^{(t)}) \\ (\sigma^2)^{(t+1)} &= \underset{\sigma^2 \in \mathbb{R}_+}{\text{argmax}}\, F(q^{(t+1)}, g^{(t+1)}, \sigma^2) \end{aligned}$$

is uniquely determined. Note that F is continuously differentiable since its parametric form (2.60) using the parametrization (2.47) is continuously differentiable in the feasibility set. Furthermore, each coordinate update is uniquely determined by Proposition 2.3.1 (see proofs in Appendix 2.7.2), under the mild assumption

$$0 \leq \sigma_1^2 < \cdots < \sigma_K^2 < \infty, \quad \pi_1, \cdots, \pi_K, \sigma^2 > 0.$$

Note that when $\pi_k = 0$, it remains 0 along the iterations. If it is initialized at $\pi_k > 0$, it can converge to $\pi_k = 0$ but remains positive along the iterations.

Thus, when initialized at $\pi_1, \cdots, \pi_K, \sigma^2 > 0$, Algorithm 1 (hence Algorithm 2) converges monotonically to a local maximum. This completes the proof. □

Proof. **of Proposition 2.3.3**

Note that the ELBO

$$F(q, g, \sigma^2) = \sum_{\boldsymbol{\gamma}} \int q(\mathbf{b}, \boldsymbol{\gamma}) \log \frac{p(\mathbf{y} \mid \mathbf{X}, \mathbf{b}, \sigma^2) p(\mathbf{b}, \boldsymbol{\gamma} \mid g, \sigma^2)}{q(\mathbf{b}, \boldsymbol{\gamma})} d\mathbf{b}$$

is maximized exactly when $q(\mathbf{b}, \boldsymbol{\gamma})$ is proportional to $p(\mathbf{y} \mid \mathbf{X}, \mathbf{b}, \sigma^2) p(\mathbf{b}, \boldsymbol{\gamma} \mid g, \sigma^2)$ as a function, that is, for any $\mathbf{b} \in \mathbb{R}^p$ and $\boldsymbol{\gamma} \in \{1, \cdots, K\}^p$. This follows from the equality condition of Jensen's inequality [83]. Note that $p(\mathbf{y} \mid \mathbf{X}, \mathbf{b}, \sigma^2)$ and $p(\mathbf{b}, \boldsymbol{\gamma} \mid g, \sigma^2)$ factorize into the product of the terms, each of which involves b_j and γ_j. This implies that the true posterior factorizes, i.e.

$$\begin{aligned} q(\mathbf{b}, \boldsymbol{\gamma}) &\propto p(\mathbf{y} \mid \mathbf{X}, \mathbf{b}, \sigma^2) p(\mathbf{b}, \boldsymbol{\gamma} \mid g, \sigma^2) \\ &\propto \prod_{j=1}^{p} \left[\exp\left(-\frac{(b_j - \mathbf{x}_j^T \mathbf{y})^2}{2\sigma^2} \right) p(b_j, \gamma_j \mid g, \sigma^2) \right]. \end{aligned}$$

Therefore, the variational approximation is exact, and fitting the multiple linear regression model is equivalent to fitting the NM model where $\mathbf{x}_1^T \mathbf{y}, \cdots, \mathbf{x}_p^T \mathbf{y}$ serve as data. □

Proof. **of Theorem 2.4.1**

We will prove a slightly more general version, Theorem 5′ below. The proof of Theorem 2.4.1 directly follows by letting $d_j = 1$ for all j, and by considering the simplified notation $\rho_{g,\sigma}$ such that

$$\rho_{g,\sigma}(b) = \sigma^2 \cdot \widetilde{\rho}_{g_\sigma,\sigma^2}(b),$$

which satisfies

$$(\rho_{g,\sigma})'(b) = S_{g,\sigma}^{-1}(b) - b.$$

This completes the proof. □

Theorem 5′. *Let $\widehat{q}$ be any local maximizer of the ELBO* (2.19) *given g and σ^2, and let $\widehat{\mathbf{b}}$ be the first moment of $\widehat{q}$. Then $\widehat{\mathbf{b}}$ is a local minimizer of the penalized linear regression objective*

$$h(\mathbf{b}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{b}\|^2 + \sum_{j=1}^{p} \sigma^2 \cdot \widetilde{\rho}_{g_\sigma,\sigma^2/d_j}(b_j), \tag{2.74}$$

where $\widetilde{\rho}$ is the penalty satisfying (2.71) *as in Lemma 2.7.2. Furthermore, if $\widehat{q}$ is a global maximizer of the ELBO, then $\widehat{\mathbf{b}}$ is a global minimizer of h.*

Proof. The proof directly follows from Proposition 2.7.1 and the definition (2.72), meaning that:

$$h(\widehat{\mathbf{b}}) = -\sigma^2 \cdot F(\widehat{\mathbf{b}}, g, \sigma^2) - \frac{n\sigma^2}{2}\log(2\pi\sigma^2).$$

Therefore, a larger F value at $\widehat{q}$ corresponds to a smaller h value at $\widehat{\mathbf{b}}$. This completes the proof. □

Lemma 2.7.3 (Property of Posterior Mean Shrinkage Operator)**.** *Suppose g be a symmetric unimodal distribution having mode at 0 and $\sigma^2 > 0$. Then $S_{g,\sigma}$ is symmetric, non-negative, non-decreasing and $S_{g,\sigma}(b) \leq b$ on $(0, \infty)$. That is, $S_{g,\sigma}$ is a shrinkage operator towards 0.*

Proof. By Khintchine's representation theorem [35], g can be represented by mixture of uniform distributions, i.e.

$$g(b) = \int_0^\infty \frac{\mathbb{I}\{|\mathbf{b}| < t\}}{2t} p(t)dt$$

for some (possibly improper) univariate mixing density $p(t)$. Let $p(b \mid t) = \mathbb{I}\{|\mathbf{b}| < t\}/(2t)$ be the Unif$(-t, t)$ density. Then we have

$$\begin{aligned} p(y \mid g, \sigma^2) &= \int p(y \mid b, \sigma^2) p(b \mid g, \sigma^2) db \\ &= \int_0^\infty \int p(y \mid b, \sigma^2) p(b \mid t) db \, p(t) dt \\ &= \int_0^\infty p(y \mid \sigma^2, t) p(t) dt \\ &= \int_0^\infty \frac{1}{2t} \left[\Phi\left(\frac{t-y}{\sigma}\right) + \Phi\left(\frac{t+y}{\sigma}\right) - 1 \right] p(t) dt. \end{aligned}$$

where Φ is the standard normal cdf. Since $\Phi(t+y) + \Phi(t-y)$ is non-increasing in $y \in (0, \infty)$ for any $t \geq 0$, $p(y \mid \sigma^2, t)$ is also non-increasing in $y \in (0, \infty)$ for any $t \geq 0$. This implies that each $p(y \mid \sigma^2, t)$ is unimodal at 0 and thus $p(y \mid g, \sigma^2)$ is also unimodal at 0 since it is a mixture of unimodal distributions at 0. Again, this implies that $S_{g,\sigma}(y) = y + \sigma^2 \ell'_{g,\sigma}(y) \leq y$ since $\ell_{g,\sigma}(y)$ is non-increasing in $y \in (0, \infty)$, where $\ell_{g,\sigma} \triangleq \log p(y \mid g, \sigma^2)$.

Also, since $p(\cdot \mid g, \sigma^2)$ is symmetric around 0, we have $S_{g,\sigma}(y) = -S_{g,\sigma}(-y)$.

Now it remains to show that $S_{g,\sigma}$ is non-decreasing on $\mathbb{R}_+$. It is easy to show that

$$p(b \mid y, \sigma^2, t) = \mathcal{TN}(b; y, \sigma^2; -t, t)$$

where $\mathcal{TN}(b; y, \sigma^2; -t, t)$ be a truncated normal density, i.e. the normal distribution with mean y and variance σ^2 is truncated to $(-t, t)$. The posterior mean of $p(b \mid y, \sigma^2, t)$ or the truncated normal distribution $\mathcal{TN}(b; y, \sigma^2; -t, t)$ is a non-decreasing function

$$\mathbb{E}_{TN(y,\sigma^2;-t,t)}(b) = y + \frac{\mathcal{N}(y+t; 0, \sigma^2) - \mathcal{N}(y-t; 0, \sigma^2)}{\Phi((t-y)/\sigma) - \Phi(-(t+y)/\sigma)} \sigma$$

of $y \in (0, \infty)$ for all $t > 0$, since we have

$$\frac{\partial}{\partial y} \mathbb{E}_{TN(y,\sigma^2;-t,t)}(b) = \frac{1}{\sigma^2} \mathrm{Var}_{TN(y,\sigma^2;-t,t)}(b) > 0.$$

Thus $S_{g,\sigma}$ is a sum of non-decreasing functions on $\mathbb{R}_+$. This completes the proof. $\square$

CHAPTER 3

BAYESIAN MODEL SELECTION WITH GRAPH STRUCTURED SPARSITY

3.1 Introduction

Bayesian model selection has been an important area of research for several decades. While the general goal is to estimate the most plausible sub-model from the posterior distribution [9, 36, 125, 20] for a wide class of learning tasks, most of the developments of Bayesian model selection have been focused on variable selection in the setting of sparse linear regression [70, 97, 60, 130, 162]. One of the main challenges of Bayesian model selection is its computational efficiency. Recently, Ročková and George [130] discovered that Bayesian variable selection in sparse linear regression can be solved by an EM algorithm [34, 113] with a closed-form update at each iteration. Compared with previous stochastic search type of algorithms such as Gibbs sampling [55, 56], this deterministic alternative greatly speeds up computation for large-scale and high-dimensional data sets.

The main thrust of this chapter is to develop of a general framework of Bayesian models that includes sparse linear regression, change-point detection, clustering and many other models as special cases. We will derive a general EM-type algorithm that efficiently explores possible candidates for Bayesian model selection. When applied to sparse linear regression, our model and algorithmic frameworks naturally recover the proposal of [130]. The general framework proposed in this chapter can be viewed as an algorithmic counterpart of the theoretical framework for Bayesian high-dimensional structured linear models in [53]. While the work [53] is focused on optimal posterior contraction rate and oracle inequalities, the current chapter pursues a general efficient and scalable computational strategy.

In order to study various Bayesian models from a unified perspective, we introduce a spike-and-slab Laplacian prior distribution on the model parameters. The new prior distribution is an extension of the classical spike-and-slab prior [109, 55, 56] for Bayesian variable selection. Our new

2. This work is in collaboration with Chao Gao [88].

definition incorporates the graph Laplacian of the underlying graph representing the model structure, and thus gives the name of the prior. Under this general framework, the problem of Bayesian model selection can be recast as selecting a subgraph of some base graph determined by the statistical task. Here, the base graph and its subgraphs represent the structures of the full model and the corresponding sub-models, respectively. Various choices of base graphs lead to specific statistical estimation problems such as sparse linear regression, clustering and change-point detection. In addition, the connection to graph algebra further allows us to build prior distributions for even more complicated models. For example, using graph products such as Cartesian product or Kronecker product [76, 96], we can construct prior distributions for biclustering models from the Laplacian of the graph products of row and column clustering structures. This leads to great flexibility in analyzing real data sets of complex structures.

Our Bayesian model selection follows the procedure of Ročková and George [130] that evaluates the posterior probabilities of sub-models computed from the solution path of the EM algorithm. However, the derivation of the EM algorithm under our general framework is indeed nontrivial task. When the underlying base graph of the model structure is a tree, the derivation of the EM algorithm is straightforward by following the arguments in [130]. On the other hand, for a general base graph that is not a tree, the arguments in [130] do not apply. To overcome this difficulty, we introduce a relaxation through the concept of effective resistance [100, 59, 141] that adapts to the underlying graphical structure of the model. The lower bound given by this relaxation is then used to derive a variational EM algorithm that works under the general framework.

Model selection with graph structured sparsity has also been studied in the frequentist literature. For example, generalized Lasso [154, 4] and its multivariate version network Lasso [69] encode the graph structured sparsity with ℓ_1 regularization. Algorithms based on ℓ_0 regularization have also been investigated recently [45, 173]. Compared with these frequentist methods, our proposed Bayesian model selection procedure tends to achieve better model selection performance in terms of false discovery proportion and power in a wide range of model scenarios, which will be shown through an extensive numerical study under various settings.

The rest of the chapter is organized as follows. In Section 3.2, we introduce the general framework of Bayesian models and discuss the spike-and-slab Laplacian prior. The EM algorithm will be derived in Section 3.3 for both the case of trees and general base graphs. In Section 3.4, we discuss how to incorporate latent variables and propose a new Bayesian clustering models under our framework. Section 3.5 introduces the techniques of graph products and several important extensions of our framework. We will also discuss a non-Gaussian spike-and-slab Laplacian prior in Section 3.6 with a natural application to reduced isotonic regression [134]. Finally, extensive simulated and real data analysis will be presented in Section 3.7.

3.2 A General Framework of Bayesian Models

In this section, we describe a general framework for building Bayesian structured models on graphs. To be specific, the prior structural assumption on the parameter $\theta \in \mathbb{R}^p$ will be encoded by a graph. Throughout the chapter, $G = (V, E)$ is an undirected graph with $V = [p]$ and some $E \subset \{(i, j) : 1 \leq i < j \leq p\}$. It is referred to as the *base graph* of the model, and our goal is to learn a sparse subgraph of G from the data. We use $p = |V|$ and $m = |E|$ for the node size and edge size of the base graph.

3.2.1 Model Description

We start with the Gaussian linear model $y|\beta, \sigma^2 \sim N(X\beta, \sigma^2 I_n)$ that models an n-dimensional observation. The design matrix $X \in \mathbb{R}^{n\times p}$ is determined by the context of the problem. Given some nonzero vector $w \in \mathbb{R}^p$, the Euclidean space $\mathbb{R}^p$ can be decomposed as a direct sum of the one-dimensional subspace spanned by w and its orthogonal complement. In other words, we can write

$$\beta = \frac{1}{\|w\|^2} ww^T\beta + \left(I_p - \frac{1}{\|w\|^2} ww^T\right)\beta.$$

The structural assumption will be imposed by a prior on the second term above. To simplify the notation, we introduce the space $\Theta_w = \left\{\theta \in \mathbb{R}^p : w^T\theta = 0\right\}$. Then, any $\beta \in \mathbb{R}^p$ can be

decomposed as $\beta = \alpha w + \theta$ for some $\alpha \in \mathbb{R}$ and $\theta \in \Theta_w$. The likelihood is thus given by

$$y|\alpha, \theta, \sigma^2 \sim N(X(\alpha w + \theta), \sigma^2 I_n). \tag{3.1}$$

The prior distribution on the vector $\alpha w + \theta$ will be specified by independent priors on α and θ. They are given by

$$\begin{aligned} \alpha|\sigma^2 &\sim N(0, \sigma^2/\nu), & (3.2)\\ \theta|\gamma, \sigma^2 &\sim p(\theta|\gamma, \sigma^2) \propto \prod_{(i,j)\in E} \exp\left(-\frac{(\theta_i - \theta_j)^2}{2\sigma^2[v_0\gamma_{ij} + v_1(1-\gamma_{ij})]}\right) \mathbb{I}\{\theta \in \Theta_w\}. & (3.3) \end{aligned}$$

Under the prior distribution, α is centered at 0 and has precision ν/σ^2. The parameter θ is modeled by a prior distribution on Θ_w that encodes a pairwise relation between θ_i and θ_j. Here, v_0 is a very small scalar and v_1 is a very large scalar. For a pair $(i, j) \in E$ in the base graph, the prior enforces the closedness between θ_i and θ_j when $\gamma_{ij} = 1$. Our goal is then to learn the most probable subgraph structure encoded by $\{\gamma_{ij}\}$, which will be estimated from the posterior distribution.

We finish the Bayesian modeling by putting priors on γ and σ^2. They are given by

$$\begin{aligned} \gamma|\eta &\sim p(\gamma|\eta) \propto \prod_{(i,j)\in E} \eta^{\gamma_{ij}}(1-\eta)^{1-\gamma_{ij}} \mathbb{I}\{\gamma \in \Gamma\}, & (3.4)\\ \eta &\sim \text{Beta}(A, B), & (3.5)\\ \sigma^2 &\sim \text{InvGamma}(a/2, b/2). & (3.6) \end{aligned}$$

Besides the standard conjugate priors on η and σ^2, the independent Bernoulli prior on γ is restricted on a set $\Gamma \subset \{0, 1\}^m$. This restriction is sometimes useful for particular models, but for now we assume that $\Gamma = \{0, 1\}^m$ until it is needed in Section 3.4.

The Bayesian model is now fully specified. The joint distribution is

$$p(y, \alpha, \theta, \gamma, \eta, \sigma^2) = p(y|\alpha, \theta, \sigma^2)p(\alpha|\sigma^2)p(\theta|\gamma, \sigma^2)p(\gamma|\eta)p(\eta)p(\sigma^2). \tag{3.7}$$

Among these distributions, the most important one is $p(\theta|\gamma, \sigma^2)$. To understand its properties, we introduce the *incidence matrix* $D \in \mathbb{R}^{m\times p}$ for the base graph $G = (V, E)$. The matrix D has entries $D_{ei} = 1$ and $D_{ej} = -1$ if $e = (i, j)$, and $D_{ek} = 0$ if $k \neq i, j$. We note that the definition of D depends on the order of edges $\{(i, j)\}$ even if G is an undirected graph. However, this does not affect any application that we will need in the chapter. We then define the Laplacian matrix

$$L_\gamma = D^T \,\text{diag}\left(v_0^{-1}\gamma + v_1^{-1}(1-\gamma)\right) D.$$

It is easy to see that L_γ is the graph Laplacian of the weighted graph with adjacency matrix $\{v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij})\}$. Thus, we can write (3.3) as

$$p(\theta|\gamma, \sigma^2) \propto \exp\left(-\frac{1}{2\sigma^2}\theta^T L_\gamma \theta\right) \mathbb{I}\{\theta \in \Theta_w\}. \tag{3.8}$$

Given its form, we name (3.8) the *spike-and-slab Laplacian prior*.

Proposition 3.2.1. *Suppose $G = (V, E)$ is a connected base graph. For any $\gamma \in \{0, 1\}^m$ and $v_0, v_1 \in (0, \infty)$, the graph Laplacian L_γ is positive semi-definite and has rank $p - 1$. The only eigenvector corresponding to its zero eigenvalue is proportional to $\mathbb{1}_p$, the vector with all entries 1. As a consequence, as long as $\mathbb{1}_p^T w \neq 0$, the spike-and-slab Laplacian prior is a non-degenerate distribution on Θ_w. Its density function with respect to the Lebesgue measure restricted to Θ_w is*

$$p(\theta|\gamma, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{(p-1)/2}} \sqrt{\det{}_w(L_\gamma)} \exp\left(-\frac{1}{2\sigma^2}\theta^T L_\gamma \theta\right) \mathbb{I}\{\theta \in \Theta_w\},$$

where $\det_w(L_\gamma)$ is the product of all nonzero eigenvalues of the positive semi-definite matrix $\left(I_p - \frac{1}{\|w\|^2} ww^T\right) L_\gamma \left(I_p - \frac{1}{\|w\|^2} ww^T\right)$.

The proposition reveals two important conditions that lead to the well-definedness of the spike-and-slab Laplacian prior: the connectedness of the base graph $G = (V, E)$ and $\mathbb{1}_p^T w \neq 0$. Without either condition, the distribution would be degenerate on Θ_w. Extensions to a base graph that is not necessarily connected is possible. We leave this task to Section 3.4 and Section 3.5, where tools

from graph algebra are introduced.

3.2.2 Examples

The Bayesian model (3.7) provides a very general framework. By choosing a different base graph $G = (V, E)$, a design matrix X, a grounding vector $w \in \mathbb{R}^p$ and a precision parameter ν, we then obtain a different model. Several important examples are given below.

Example 3.2.1 (Sparse linear regression)**.** *The sparse linear regression model $y|\theta, \sigma^2 \sim N(X\theta, \sigma^2 I_n)$ is a special case of (3.1). To put it into the general framework, we can expand the design matrix $X \in \mathbb{R}^{n\times p}$ and the regression vector $\theta \in \mathbb{R}^p$ by $[0_n, X] \in \mathbb{R}^{n\times(p+1)}$ and $[\theta_0; \theta] \in \mathbb{R}^{p+1}$. With the grounding vector $w = [1; 0_p]$, the sparse linear regression model can be recovered from (3.1). For the prior distribution, the base graph G consists of nodes $V = \{0, 1, ..., p\}$ and edges $\{(0, i) : i \in [p]\}$. We set $\nu = \infty$, so that $\theta_0 = 0$ with prior probability one. Then, (3.3) is reduced to*

$$\theta|\gamma, \sigma^2 \sim p(\theta|\gamma, \sigma^2) \propto \prod_{i=1}^{p} \exp\left(-\frac{\theta_i^2}{2\sigma^2[v_0\gamma_{0i} + v_1(1-\gamma_{0i})]}\right).$$

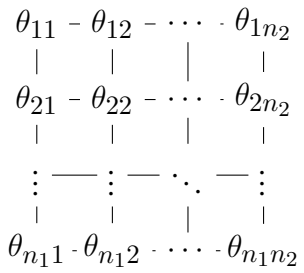
That is, $\theta_i|\gamma, \sigma^2 \sim N(0, \sigma^2[v_0\gamma_{0i} + v_1(1-\gamma_{0i})])$ independently for all $i \in [n]$. This is recognized as the spike-and-slab Gaussian prior for Bayesian sparse linear regression considered by [55, 56, 130].

Example 3.2.2 (Change-point detection)**.** *Set $n = p$, $X = I_n$, and $w = \mathbb{1}_n$. We then have $y_i|\theta, \sigma^2 \sim N(\alpha + \theta_i, \sigma^2)$ independently for all $i \in [n]$ from (3.1). For the prior distribution on α and θ, we consider $\nu = 0$ and a one-dimensional chain graph $G = (V, E)$ with $E = \{(i, i+1) : i \in [n-1]\}$. This leads to a flat prior on α, and the prior on θ is given by*

$$\theta|\gamma, \sigma^2 \sim p(\theta|\gamma, \sigma^2) \propto \prod_{i=1}^{n-1} \exp\left(-\frac{(\theta_i - \theta_{i+1})^2}{2\sigma^2[v_0\gamma_{i,i+1} + v_1(1-\gamma_{i,i+1})]}\right) \mathbb{I}\{\mathbb{1}_p^T\theta = 0\}.$$

A more general change-point model on a tree can also be obtained by constructing a tree base graph G.

Example 3.2.3 (Two-dimensional image denoising)**.** *Consider a rectangular set of observations $y \in \mathbb{R}^{n_1 \times n_2}$. With the same construction in Example 3.2.2 applied to* vec(y)*, we obtain $y_{ij}|\theta, \sigma^2 \sim N(\alpha + \theta_{ij}, \sigma^2)$ independently for all $(i,j) \in [n_1] \times [n_2]$ from (3.1). To model images, we consider a prior distribution that imposes closedness to nearby pixels. Consider $\nu = 0$ and a base graph $G = (V, E)$ shown in the picture below.*



We then obtain a flat prior on α, and

$$\theta|\gamma, \sigma^2 \sim p(\theta|\gamma, \sigma^2) \propto \prod_{(ik,jl)\in E} \exp\left(-\frac{(\theta_{ik} - \theta_{jl})^2}{2\sigma^2[v_0\gamma_{ik,jl} + v_1(1-\gamma_{ik,jl})]}\right) \mathbb{I}\{\mathbb{1}_{n_1}^T \theta \mathbb{1}_{n_2} = 0\}.$$

Note that G is not a tree in this case.

3.3 EM Algorithm

In this section, we will develop efficient EM algorithms for the general model. It turns out that the bottleneck is the computation of $\det_w(L_\gamma)$ given some $\gamma \in \{0,1\}^m$.

Lemma 3.3.1. *Let* spt(G) *be the set of all spanning trees of G. Then*

$$\det_w(L_\gamma) = \frac{(\mathbb{1}_p^T w)^2}{\|w\|^2} \sum_{T\in \text{spt}(G)} \prod_{(i,j)\in T} \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij})\right].$$

In particular, if G is a tree, then $\det_w(L_\gamma) = \frac{(\mathbb{1}_p^T w)^2}{\|w\|^2} \prod_{(i,j)\in E} \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij})\right]$.

The lemma suggests that the hardness of computing $\det_w(L_\gamma)$ depends on the number of spanning trees of the base graph G. When the base graph is a tree, $\det_w(L_\gamma)$ is factorized over the edges

of the tree, which greatly simplifies the derivation of the algorithm. We will derive a closed-form EM algorithm in Section 3.3.1 when G is a tree, and the algorithm for a general G will be given in Section 3.3.2.

3.3.1 The Case of Trees

We treat γ as latent. Our goal is to maximize the marginal distribution after integrating out the latent variables. That is,

$$\max_{\alpha,\theta\in\Theta_w,\eta,\sigma^2} \log \sum_{\gamma} p(y,\alpha,\theta,\gamma,\eta,\sigma^2), \tag{3.9}$$

where $p(y,\alpha,\theta,\gamma,\eta,\sigma^2)$ is given by (3.7). Since the summation over γ is intractable, we consider an equivalent form of (3.9), which is

$$\max_{q} \max_{\alpha,\theta\in\Theta_w,\eta,\sigma^2} \sum_{\gamma} q(\gamma) \log \frac{p(y,\alpha,\theta,\gamma,\eta,\sigma^2)}{q(\gamma)}. \tag{3.10}$$

Then, the EM algorithm is equivalent to iteratively updating $q, \alpha, \theta \in \Theta_w, \eta, \sigma^2$ [113].

Now we illustrate the EM algorithm that solves (3.10). The E-step is to update $q(\gamma)$ given the previous values of θ, η, σ. In view of (3.7), we have

$$q^{\text{new}}(\gamma) \propto p(y,\alpha,\theta,\gamma,\eta,\sigma^2) \propto p(\theta|\gamma,\sigma^2)p(\gamma|\eta). \tag{3.11}$$

According to (3.3.1), $p(\theta|\gamma,\sigma^2)$ can be factorized when the base graph $G = (V,E)$ is a tree. Therefore, with a simpler notation $q_{ij} = q(\gamma_{ij} = 1)$, we have $q^{\text{new}}(\gamma) = \prod_{(i,j)\in E}(q_{ij}^{\text{new}})^{\gamma_{ij}}(1 - q_{ij}^{\text{new}})^{1-\gamma_{ij}}$, where

$$q_{ij}^{\text{new}} = \frac{\eta\phi(\theta_i - \theta_j; 0, \sigma^2 v_0)}{\eta\phi(\theta_i - \theta_j; 0, \sigma^2 v_0) + (1-\eta)\phi(\theta_i - \theta_j; 0, \sigma^2 v_1)}. \tag{3.12}$$

Here, $\phi(\cdot;\mu,\sigma^2)$ stands for the density function of $N(\mu,\sigma^2)$.

To derive the M-step, we introduce the following function

$$F(\alpha, \theta; q) = \|y - X(\alpha w + \theta)\|^2 + \nu\alpha^2 + \theta^T L_q \theta, \tag{3.13}$$

where L_q is obtained by replacing γ with q in the definition of the graph Laplacian L_γ. The M-step consists of the following three updates,

$$\begin{aligned}
(\alpha^{\text{new}}, \theta^{\text{new}}) &= \underset{\alpha, \theta \in \Theta_w}{\text{argmin}}\, F(\alpha, \theta; q^{\text{new}}), && (3.14)\\
(\sigma^2)^{\text{new}} &= \underset{\sigma^2}{\text{argmin}} \left[\frac{F(\alpha^{\text{new}}, \theta^{\text{new}}; q^{\text{new}}) + b}{2\sigma^2} + \frac{p + n + a + 2}{2} \log(\sigma^2) \right], && (3.15)\\
\eta^{\text{new}} &= \underset{\eta}{\text{argmax}} \left[(A - 1 + q_{\text{sum}}^{\text{new}}) \log \eta + (B - 1 + p - 1 - q_{\text{sum}}^{\text{new}}) \log(1 - \eta) \right]. && (3.16)
\end{aligned}$$

where the notation $q_{\text{sum}}^{\text{new}}$ stands for $\sum_{(i,j)\in E} q_{ij}^{\text{new}}$. While (3.14) is a simple quadratic programming, (3.15) and (3.16) have closed forms, which are given by

$$(\sigma^2)^{\text{new}} = \frac{F(\alpha^{\text{new}}, \theta^{\text{new}}; q^{\text{new}}) + b}{p + n + a + 2} \quad \text{and} \quad \eta^{\text{new}} = \frac{A - 1 + q_{\text{sum}}^{\text{new}}}{A + B + p - 3}. \tag{3.17}$$

We remark that the EMVS algorithm [130] is a special case for the sparse linear regression problem discussed in Example 3.2.1. When G is a tree, the spike-and-slab graph Laplacian prior (3.8) is proportional to the product of individual spike-and-slab priors

$$p(\theta|\gamma, \sigma^2) \propto \prod_{(i,j)\in E} \exp\left(-\frac{(\theta_i - \theta_j)^2}{2\sigma^2[v_0\gamma_{ij} + v_1(1 - \gamma_{ij})]} \right),$$

supported on Θ_w, as we have seen in Example 3.2.1 and 3.2.2. In this case, the above EM algorithm we have developed can also be extended to models with alternative prior distributions, such as the spike-and-slab Lasso prior [131] and the finite normal mixture prior [144].

3.3.2 General Graphs

When the base graph G is not a tree, the E-step becomes computationally infeasible due to the lack of separability of $p(\theta|\gamma, \sigma^2)$ in γ. In fact, given the form of the density function in Proposition 3.2.1, the main problem lies in the term $\sqrt{\det_w(L_\gamma)}$, which cannot be factorized over $(i,j) \in E$ when the base graph $G = (V, E)$ is not a tree (Lemma 3.3.1). To overcome the difficulty, we consider optimizing a lower bound of the objective function (3.10). This means we need to find a good lower bound for $\log \det_w(L_\gamma)$. Similar techniques are also advocated in the context of learning exponential family graphical models [159].

By Lemma 3.3.1, we can write

$$\log \det_w(L_\gamma) = \log \sum_{T \in \text{spt}(G)} \prod_{(i,j)\in T} \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right] + \log \frac{(\mathbb{1}_p^T w)^2}{\|w\|^2}. \tag{3.18}$$

We only need to lower bound the first term on the right hand side of the equation above, because the second term is independent of γ. By Jensen's inequality, for any non-negative sequence $\{\lambda(T)\}_{T\in\text{spt}(G)}$ such that $\sum_{T\in\text{spt}(G)} \lambda(T) = 1$, we have

$$\begin{aligned}
& \log \sum_{T \in \text{spt}(G)} \prod_{(i,j)\in T} \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right] \\
\geq & \sum_{T \in \text{spt}(G)} \lambda(T) \log \prod_{(i,j)\in T} \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right] - \sum_{T \in \text{spt}(G)} \lambda(T) \log \lambda(T) \\
= & \sum_{(i,j)\in E} \left(\sum_{T \in \text{spt}(G)} \lambda(T) \mathbb{I}\{(i,j) \in T\} \right) \log \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right] \\
& - \sum_{T \in \text{spt}(G)} \lambda(T) \log \lambda(T).
\end{aligned}$$

One of the most natural choices of the weights $\{\lambda(T)\}_{T\in\text{spt}(G)}$ is the uniform distribution

$$\lambda(T) = \frac{1}{|\text{spt}(G)|}.$$

This leads to the following lower bound

$$\begin{aligned} & \log \sum_{T \in \text{spt}(G)} \prod_{(i,j) \in T} \left[v_0^{-1} \gamma_{ij} + v_1^{-1} (1 - \gamma_{ij}) \right] \\ \geq & \sum_{(i,j) \in E} r_{ij} \log \left[v_0^{-1} \gamma_{ij} + v_1^{-1} (1 - \gamma_{ij}) \right] + \log |\text{spt}(G)|, \end{aligned} \tag{3.19}$$

where

$$r_{ij} = \frac{1}{|\text{spt}(G)|} \sum_{T \in \text{spt}(G)} \mathbb{I}\{(i,j) \in T\}. \tag{3.20}$$

The quantity r_{ij} defined in (3.20) is recognized as the *effective resistance* between the ith and the jth nodes [100, 59]. Given a graph, we can treat each edge as a resistor with resistance 1. Then, the effective resistance between the ith and the jth nodes is the resistance between i and j given by the whole graph. That is, if we treat the entire graph as a resistor. Let L be the (unweighted) Laplacian matrix of the base graph $G = (V, E)$, and L^+ its pseudo-inverse. Then, an equivalent definition of (3.20) is given by the formula

$$r_{ij} = (e_i - e_j)^T L^+ (e_i - e_j),$$

where e_j is the basis vector with the ith entry 1 and the remaining entries 0. Therefore, computation of the effective resistance can leverage fast Laplacian solvers in the literature [142, 98]. Some important examples of effective resistance are listed below:

- When G is the complete graph of size p, then $r_{ij} = 2/p$ for all $(i, j) \in E$.
- When G is the complete bipartite graph of sizes p and k, then $r_{ij} = \frac{p+k-1}{pk}$ for all $(i, j) \in E$.
- When G is a tree, then $r_{ij} = 1$ for all $(i, j) \in E$.
- When G is a two-dimensional grid graph of size $n_1 \times n_2$, then $r_{ij} \in [0.5, 0.75]$ depending on how close the edge (i, j) is from its closest corner.
- When G is a lollipop graph, the conjunction of a linear chain with size p and a complete graph

with size k, then $r_{ij} = 1$ or $2/k$ depending on whether the edge (i, j) belongs to the chain or the complete graph.

By (3.19), we obtain the following lower bound for the objective function (3.10),

$$\max_{q} \max_{\alpha,\theta\in\Theta_w,\eta,\sigma^2} \sum_{\gamma} q(\gamma) \log \frac{p(y|\alpha,\theta,\sigma^2)p(\alpha|\sigma^2)\widetilde{p}(\theta|\gamma,\sigma^2)p(\gamma|\eta)p(\eta)p(\sigma^2)}{q(\gamma)}, \tag{3.21}$$

where the formula of $\widetilde{p}(\theta|\gamma,\sigma^2)$ is obtained by applying the lower bound (3.19) in the formula of $p(\theta|\gamma,\sigma^2)$ in Proposition 3.2.1. Since $\widetilde{p}(\theta|\gamma,\sigma^2)$ can be factorized over $(i,j) \in E$, the E-step is given by $q^{\text{new}}(\gamma) = \prod_{(i,j)\in E}(q_{ij}^{\text{new}})^{\gamma_{ij}}(1-q_{ij}^{\text{new}})^{1-\gamma_{ij}}$, where

$$q_{ij}^{\text{new}} = \frac{\eta v_0^{-\frac{r_{ij}}{2}} e^{-\frac{1}{2\sigma^2 v_0}(\theta_i-\theta_j)^2}}{\eta v_0^{-\frac{r_{ij}}{2}} e^{-\frac{1}{2\sigma^2 v_0}(\theta_i-\theta_j)^2} + (1-\eta) v_1^{-\frac{r_{ij}}{2}} e^{-\frac{1}{2\sigma^2 v_1}(\theta_i-\theta_j)^2}}. \tag{3.22}$$

Observe that the lower bound (3.19) is independent of $\alpha, \theta, \eta, \sigma^2$, and thus the M-step remains the same as in the case of a tree base graph. The formulas are given by (3.14)-(3.16), except that (3.16) needs to be replaced by

$$\eta^{\text{new}} = \frac{A - 1 + q_{\text{sum}}^{\text{new}}}{A + B + m - 2}.$$

The EM algorithm for a general base graph can be viewed as a natural extension of that of a tree base graph. When $G = (V, E)$ is a tree, it is easy to see from the formula (3.20) that $r_{ij} = 1$ for all $(i, j) \in E$. In this case, the E-step (3.22) is reduced to (3.12), and the inequality (3.19) becomes an equality.

3.3.3 Bayesian Model Selection

The output of the EM algorithm $\widehat{q}(\gamma)$ can be understood as an estimator of the posterior distribution $p(\gamma|\widehat{\alpha},\widehat{\theta},\widehat{\sigma}^2,\widehat{\eta})$, where $\widehat{\alpha},\widehat{\theta},\widehat{\sigma}^2,\widehat{\eta}$ are obtained from the M-step. Then, we get a subgraph according to the thresholding rule $\widehat{\gamma}_{ij} = \mathbb{I}\{\widehat{q}_{ij} \geq 1/2\}$. It can be understood as a model learned from the data. The sparsity of the model critically depends on the values of v_0 and v_1 in the spike-and-slab

Laplacian prior. With a fixed large value of v_1, we can obtain the solution path of $\widehat{\gamma} = \widehat{\gamma}(v_0)$ by varying v_0 from 0 to v_1. The question then is how to select the best model along the solution path of the EM algorithm.

The strategy suggested by [130] is to calculate the posterior score $p(\gamma|y)$ with respect to the Bayesian model of $v_0 = 0$. While the meaning of $p(\gamma|y)$ corresponding to $v_0 = 0$ is easily understood for the sparse linear regression setting in [130], it is less clear for a general base graph $G = (V, E)$.

In order to define a version of (3.7) for $v_0 = 0$, we need to introduce the concept of *edge contraction*. Given a $\gamma \in \{0,1\}^m$, the graph corresponding to the adjacency matrix γ induces a partition of disconnected components $\{\mathcal{C}_1, ..., \mathcal{C}_s\}$ of $[p]$. In other words, $\{i, j\} \subset \mathcal{C}_l$ for some $l \in [s]$ if and only if there is some path between i and j in the graph γ. For notational convenience, we define a vector $z \in [s]^n$ so that $z_i = l$ if and only if $i \in \mathcal{C}_l$. A membership matrix $Z_\gamma \in \{0,1\}^{p\times s}$ is defined with its (i, l)th entry being the indicator $\mathbb{I}\{z_i = l\}$.

We let $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ be a graph obtained from the base graph $G = (V, E)$ after the operation of edge contraction. In other words, every node in $\widetilde{G}$ is obtained by combining nodes in G according to the partition of $\{\mathcal{C}_1, ..., \mathcal{C}_s\}$. To be specific, $\widetilde{V} = [s]$, and $(k, l) \in \widetilde{E}$ if and only if there exists some $i \in \mathcal{C}_k$ and some $j \in \mathcal{C}_l$ such that $(i, j) \in E$.

Now we are ready to define a limiting version of (3.3) as $v_0 \to 0$. Let $\widetilde{L}_\gamma = D^T \operatorname{diag}(v_1^{-1}(1 - \gamma))D$, which is the graph Laplacian of the weighted graph with adjacency matrix $\{v_1^{-1}(1 - \gamma_{ij})\}$. Then, define

$$p(\widetilde{\theta}|\gamma, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{(s-1)/2}} \sqrt{\det{}_{Z_\gamma^T w}(Z_\gamma^T \widetilde{L}_\gamma Z_\gamma)} \exp\left(-\frac{\widetilde{\theta}^T Z_\gamma^T \widetilde{L}_\gamma Z_\gamma \widetilde{\theta}}{2\sigma^2}\right) \mathbb{I}\{\widetilde{\theta} \in \Theta_{Z_\gamma^T w}\}. \tag{3.23}$$

With $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ standing for the contracted base graph, the prior distribution (3.23) can also be written as

$$p(\widetilde{\theta}|\gamma, \sigma^2) \propto \exp\left(-\sum_{(k,l)\in\widetilde{E}} \frac{\omega_{kl}(\widetilde{\theta}_k - \widetilde{\theta}_l)^2}{2\sigma^2 v_1}\right) \mathbb{I}\{\widetilde{\theta} \in \Theta_{Z_\gamma^T w}\}, \tag{3.24}$$

where $\omega_{kl} = \sum_{(i,j)\in E} \mathbb{I}\{z(i) = k, z(j) = l\}$, which means that the edges $\{(i,j)\}_{z(i)=k,z(j)=l}$ in the base graph $G = (V, E)$ are contracted as a new edge (k, l) in $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ with ω_{kl} as the weight.

Proposition 3.3.1. *Suppose $G = (V, E)$ is connected and $\mathbb{1}_p^T w \neq 0$. Let Z_γ be the membership matrix defined as above. Then for any $\gamma \in \{0, 1\}^m$, (3.23) is a well-defined density function on the $(s-1)$-dimensional subspace $\{\widetilde{\theta} \in \mathbb{R}^s : w^T Z_\gamma \widetilde{\theta} = 0\}$. Moreover, for an arbitrary design matrix $X \in \mathbb{R}^{n\times p}$, the distribution of θ that follows (3.3) weakly converges to that of $Z_\gamma\widetilde{\theta}$ as $v_0 \to 0$.*

Motivated by Proposition 3.3.1, a limiting version of (3.7) for $v_0 = 0$ is defined as follows,

$$y|\alpha, \widetilde{\theta}, \gamma, \sigma^2 \sim N(X(\alpha w + Z_\gamma\widetilde{\theta}), \sigma^2 I_n). \tag{3.25}$$

Then, $p(\widetilde{\theta}|\gamma, \sigma^2)$ is given by (3.23), and $p(\alpha|\sigma^2)$, $p(\gamma|\eta)$, $p(\eta)$, $p(\sigma^2)$ are specified in (3.2) and (3.4)-(3.6). The posterior distribution of γ has the formula

$$\begin{aligned} p(\gamma|y) &\propto \int\int\int\int p(y, \alpha, \widetilde{\theta}, \gamma, \eta, \sigma^2) d\alpha d\widetilde{\theta} d\eta d\sigma^2 \\ &= \int p(\sigma^2) \int p(\alpha|\sigma^2) \int p(y|\alpha, \widetilde{\theta}, \gamma, \sigma^2) p(\widetilde{\theta}|\gamma, \sigma^2) d\widetilde{\theta} d\alpha d\sigma^2 \int p(\gamma|\eta) p(\eta) d\eta. \end{aligned}$$

A standard calculation using conjugacy gives

$$\begin{aligned} p(\gamma|y) \propto & \left(\frac{\det_{Z_\gamma^T w}(Z_\gamma^T \widetilde{L}_\gamma Z_\gamma)}{\det_{Z_\gamma^T w}(Z_\gamma^T (X^T X + \widetilde{L}_\gamma) Z_\gamma)} \right)^{1/2} \left(\frac{\nu}{\nu + w^T X^T (I_n - R_\gamma) X w} \right)^{1/2} \\ & \times \left(y^T (I_n - R_\gamma) y - \frac{|w^T X^T (I_n - R_\gamma) y|^2}{\nu + w^T X^T (I_n - R_\gamma) X w} + b \right)^{-\frac{n+a}{2}} \\ & \times \frac{\text{Beta}\left(\sum_{(i,j)\in E} \gamma_{ij} + A - 1, \sum_{(i,j)\in E} (1 - \gamma_{ij}) + B - 1 \right)}{\text{Beta}(A, B)}, \end{aligned} \tag{3.26}$$

where

$$R_\gamma = X Z_\gamma (Z_\gamma^T (X^T X + \widetilde{L}_\gamma) Z_\gamma)^{-1} Z_\gamma^T X^T.$$

This defines the model selection score $g(\gamma) = \log p(\gamma|y)$ up to a universal additive constant. The

Bayesian model selection procedure evaluates $g(\gamma)$ on the solution path $\{\widehat{\gamma}(v_0)\}_{0<v_0\leq v_1}$ and selects the best model with the highest value of $g(\gamma)$.

3.3.4 Summary of Our Approach

The parameter v_0 plays a critical role in our Bayesian model selection procedure. Recall that the joint distribution of our focus has the expression

$$p(y|\theta)p_{v_0}(\theta|\gamma)p(\gamma), \tag{3.27}$$

where $p(y|\theta)$ is the linear model parametrized by θ, $p_{v_0}(\theta|\gamma)$ is the spike-and-slab prior with tuning parameter v_0, and $p(\gamma)$ is the prior on the graph[1]. The tuning parameter v_0 is the variance of the spike component of the prior. Different choices of v_0 are used as different components in our entire model selection procedure.

1. The ideal choice of $v_0 = 0$ models exact sparsity in the sense that $\gamma_{ij} = 1$ implies $\theta_i = \theta_j$. In this case, the exact posterior

$$p_{v_0=0}(\gamma|y) \propto \int p(y|\theta)p_{v_0=0}(\theta|\gamma)p(\gamma)d\theta$$

   can be calculated according to the formulas that we derive in Section 3.3.3. Then the ideal Bayes model selection procedure would be the one that maximizes the posterior $p_{v_0=0}(\gamma|y)$ over all γ. However, since this would require evaluating $p_{v_0=0}(\gamma|y)$ for exponentially many γ's, it is sensible to maximize $p_{v_0=0}(\gamma|y)$ only over a carefully chosen subset of γ that has a reasonable size for computational efficiency.

2. The choice of $v_0 > 0$ models approximate sparsity in the sense that γ_{ij} implies $\theta_i \approx \theta_j$. Though $v_0 > 0$ does not offer interpretation of exact sparsity, a nonzero v_0 leads to efficient

1. We have ignored other parameters such as α, η, σ^2 in order to make the discussion below clear and concise.

computation via the EM algorithm. That is, for a $v_0 > 0$, we can maximize

$$\max_q \max_\theta \sum_\gamma q(\gamma) \log \frac{p(y|\theta)p_{v_0}(\theta|\gamma)p(\gamma)}{q(\gamma)},$$

which is the objective function of EM. Denote the output of the algorithm by $q_{v_0}(\gamma) = \prod_{ij} q_{ij,v_0}$, we then obtain our model by $\widehat{\gamma}_{ij}(v_0) = \mathbb{I}\{q_{ij,v_0} > 0.5\}$. As we vary v_0 on a grid from 0 to v_1, we obtain a path of models $\{\widehat{\gamma}(v_0)\}_{0<v_0\leq v_1}$. It covers models that ranges from very parsimonious ones to the fully saturated one.

The proposed model selection procedure in Section 3.3.3 is

$$\max_{\gamma\in\{\widehat{\gamma}(v_0)\}_{0<v_0\leq v_1}} p_{v_0=0}(\gamma|y), \tag{3.28}$$

where $p_{v_0=0}(\gamma|y) \propto \int p(y|\theta)p_{v_0=0}(\theta|\gamma)p(\gamma)d\theta$. That is, we optimize the posterior distribution $p_{v_0=0}(\gamma|y)$ *only* over the EM solution path. The best one among all the candidate models will be selected according to $p_{v_0=0}(\gamma|y)$, which is the exact/full posterior of γ. There are two ways to interpret our model selection procedure (3.28):

1. The procedure (3.28) can be understood as a computationally efficient approximation strategy to the ideal Bayes model selection procedure $\max_\gamma p_{v_0=0}(\gamma|y)$ that is infeasible to compute. The EM algorithm with various choices of v_0 simply provides a short list of candidate models. From this perspective, the proposed procedure (3.28) is fully Bayes.

2. The procedure (3.28) can be also understood as a method for selecting the tuning parameter v_0, because the maximizer of (3.28) must be in the form of $\widehat{\gamma}(\widehat{v}_0)$ for some data-driven $\widehat{v}_0$. In this way, the solution $\widehat{\gamma} = \widehat{\gamma}(\widehat{v}_0)$ also has a non-Bayesian interpretation since it is obtained by post-processing the EM solution with the tuning parameter $\widehat{v}_0$ selected by (3.28). In this regard, $\widehat{\gamma}$ also can be thought of as an empirical Bayes estimator.

In summary, our proposed procedure (3.28) is motivated by both statistical and computational considerations.

3.4 Clustering: A New Deal

3.4.1 A Multivariate Extension

Before introducing our new Bayesian clustering model, we need a multivariate extension of the general framework (3.7) to model a matrix observation $y \in \mathbb{R}^{n\times d}$. With a design matrix $X \in \mathbb{R}^{n\times p}$, the dimension of θ is now $p \times d$. We denote the ith row of θ by θ_i. With the grounding vector $w \in \mathbb{R}^p$, the distribution $p(y|\alpha, \theta, \sigma^2)p(\alpha|\sigma^2)p(\theta|\gamma, \sigma^2)$ is given by

$$
\begin{aligned}
y|\alpha, \theta, \sigma^2 &\sim N(X(w\alpha^T + \theta), \sigma^2 I_n \otimes I_d), && (3.29)\\
\alpha|\sigma^2 &\sim N\left(0, \frac{\sigma^2}{\nu} I_d\right), && (3.30)\\
\theta|\gamma, \sigma^2 &\sim p(\theta|\gamma, \sigma^2) \propto \prod_{(i,j)\in E} \exp\left(-\frac{\|\theta_i - \theta_j\|^2}{2\sigma^2[v_0\gamma_{ij} + v_1(1-\gamma_{ij})]}\right) \mathbb{I}\{\theta \in \Theta_w\}, && (3.31)
\end{aligned}
$$

where $\Theta_w = \{\theta \in \mathbb{R}^{p\times d} : w^T\theta = 0\}$. The prior distributions on γ, η, σ^2 are the same as (3.4)-(3.6). Moreover, the multivariate spike-and-slab Laplacian prior (3.31) is supported on a $d(p-1)$-dimensional subspace Θ_w, and is well-defined as long as $\mathbb{1}_p^T w \neq 0$ for the same reason stated in Proposition 3.2.1.

The multivariate extension can be understood as the task of learning d individual graphs for each column of θ. Instead of modeling the d graph separately by $\gamma^{(1)}, ..., \gamma^{(d)}$ using (3.7), we assume the d columns of θ share the same structure by imposing the condition $\gamma^{(1)} = ... = \gamma^{(d)}$.

An immediate example is a Bayesian multitask learning problem with group sparsity. It can be viewed as a multivariate extension of Example 3.2.1. With the same argument in Example 3.2.1, (3.29)-(3.31) is specialized to

$$
\begin{aligned}
y|\theta, \sigma^2 &\sim N(X\theta, \sigma^2 I_n \otimes I_d),\\
\theta|\gamma, \sigma^2 &\sim p(\theta|\gamma, \sigma^2) \propto \prod_{i=1}^{p} \exp\left(-\frac{\|\theta_i\|^2}{2\sigma^2[v_0\gamma_i + v_1(1-\gamma_i)]}\right).
\end{aligned}
$$

To close this subsection, let us mentions that the model (3.29)-(3.31) can be easily modified to accommodate a heteroscedastic setting. For example, one can replace the $\sigma^2 I_n \otimes I_d$ in (3.29) by a more general $I_n \otimes \text{diag}(\sigma_1^2, ..., \sigma_d^2)$, and then make corresponding changes to (3.30) and (3.31) as well.

3.4.2 Model Description

Consider the likelihood

$$y|\alpha, \theta, \sigma^2 \sim N(\mathbb{1}_n \alpha^T + \theta, \sigma^2 I_n \otimes I_d), \tag{3.32}$$

with the prior distribution of $\alpha|\sigma^2$ specified by (3.30). The clustering model uses the following form of (3.31),

$$p(\theta, \mu|\gamma, \sigma^2) \propto \prod_{i=1}^{n} \prod_{j=1}^{k} \exp\left(-\frac{\|\theta_i - \mu_j\|^2}{2\sigma^2[v_0\gamma_{ij} + v_1(1-\gamma_{ij})]}\right) \mathbb{I}\{\mathbb{1}_n^T \theta = 0\}. \tag{3.33}$$

Here, both vectors θ_i and μ_j are in $\mathbb{R}^d$. The prior distribution (3.33) can be derived from (3.31) by replacing θ in (3.31) with (θ, μ) and specifying the base graph as a complete bipartite graph between θ and μ. We impose the restriction that

$$\sum_{j=1}^{k} \gamma_{ij} = 1, \tag{3.34}$$

for all $i \in [n]$. Then, $\mu_1, ..., \mu_k$ are latent variables that can be interpreted as the clustering centers, and each θ_i is connected to one of the clustering centers.

To fully specify the clustering model, the prior distribution of γ is given by (3.4) with Γ being the set of all $\{\gamma_{ij}\}$ that satisfies (3.34). Equivalently,

$$(\gamma_{i1}, ..., \gamma_{ik}) \sim \text{Uniform}(\{e_j\}_{j=1}^k), \tag{3.35}$$

independently for all $i \in [n]$, where e_j is a vector with 1 on the jth entry and 0 elsewhere. Finally,

the prior of σ^2 is given by (3.6).

3.4.3 EM Algorithm

The EM algorithm can be derived by following the idea developed in Section 3.3.2. In the current setting, the lower bound (3.19) becomes

$$\begin{aligned} & \log \sum_{T \in \text{spt}(K_{n,k})} \prod_{i=1}^{n} \prod_{j=1}^{k} [v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij})] \\ \geq & \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij} \log \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right] + \log |\text{spt}(K_{n,k})|, \end{aligned} \tag{3.36}$$

where $K_{n,k}$ is the complete bipartite graph. By symmetry, the effective resistance $r_{ij} = r$ is a constant independent of (i, j). Thus, (3.36) can be written as

$$r \sum_{i=1}^{n}\sum_{j=1}^{k} \log \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right] + \log |\text{spt}(K_{n,k})| \tag{3.37}$$

$$\begin{aligned} = & \; r \log(v_0^{-1}) \sum_{i=1}^{n}\sum_{j=1}^{k} \gamma_{ij} + r \log(v_1^{-1}) \sum_{i=1}^{n}\sum_{j=1}^{k} (1-\gamma_{ij}) + \log |\text{spt}(K_{n,k})| \\ = & \; rn \log(v_0^{-1}) + rn(k-1) \log(v_1^{-1}) + \log |\text{spt}(K_{n,k})|, \end{aligned} \tag{3.38}$$

where the last equality is derived from (3.34). Therefore, for the clustering model, the lower bound (3.19) is a constant independent of $\{\gamma_{ij}\}$. As a result, the lower bound of the objective function of the EM algorithm becomes

$$\sum_{\gamma} q(\gamma) \log \frac{p(y|\alpha, \theta, \sigma^2) p(\alpha|\sigma^2) \widetilde{p}(\theta, \mu|\gamma, \sigma^2) p(\gamma) p(\sigma^2)}{q(\gamma)}, \tag{3.39}$$

with

$$\widetilde{p}(\theta, \mu|\gamma, \sigma^2) = \text{const} \times \frac{1}{(2\pi\sigma^2)^{(n+k-1)d/2}} \prod_{i=1}^{n} \prod_{j=1}^{k} \exp\left(-\frac{\|\theta_i - \mu_j\|^2}{2\sigma^2 [v_0\gamma_{ij} + v_1(1-\gamma_{ij})]} \right),$$

and the algorithm is to maximize (3.39) over $q, \alpha, \theta \in \{\theta : \mathbb{1}_n^T \theta = 0\}$ and σ^2.

Maximizing (3.39) over q, we obtain the E-step as

$$q_{ij}^{\text{new}} = \frac{\exp\left(-\frac{\|\theta_i - \mu_j\|^2}{2\sigma^2 \bar{v}}\right)}{\sum_{l=1}^{k} \exp\left(-\frac{\|\theta_i - \mu_l\|^2}{2\sigma^2 \bar{v}}\right)}, \tag{3.40}$$

independently for all $i \in [n]$, where $\bar{v}^{-1} = v_0^{-1} - v_1^{-1}$, and we have used the notation

$$q_{ij} = q\left((\gamma_{i1}, ..., \gamma_{ik}) = e_j\right).$$

It is interesting to note that the E-step only depends on v_0 and v_1 through $\bar{v}$. Maximizing (3.39) over $\alpha, \theta \in \{\theta : \mathbb{1}_n^T \theta = 0\}, \mu, \sigma^2$, we obtain the M-step as

$$\begin{aligned} (\alpha^{\text{new}}, \theta^{\text{new}}, \mu^{\text{new}}) &= \underset{\alpha, \mathbb{1}_n^T \theta = 0, \mu}{\operatorname{argmin}} \; F(\alpha, \theta, \mu; q^{\text{new}}), \\ (\sigma^2)^{\text{new}} &= \frac{F(\alpha^{\text{new}}, \theta^{\text{new}}, \mu^{\text{new}}; q^{\text{new}}) + b}{(2n + k)d + a + 2}, \end{aligned} \tag{3.41}$$

where

$$F(\alpha, \theta, \mu; q) = \|y - \mathbb{1}_n \alpha^T - \theta\|_{\text{F}}^2 + \nu \|\alpha\|^2 + \sum_{i=1}^{n} \sum_{j=1}^{k} \left(\frac{q_{ij}}{v_0} + \frac{1 - q_{ij}}{v_1}\right) \|\theta_i - \mu_j\|^2.$$

Note that for all θ such that $\mathbb{1}_n^T \theta = 0$, we have

$$\|y - \mathbb{1}_n \alpha^T - \theta\|_{\text{F}}^2 = \|n^{-1} \mathbb{1}_n \mathbb{1}_n^T y - \mathbb{1}_n \alpha^T\|_{\text{F}}^2 + \|y - n^{-1} \mathbb{1}_n \mathbb{1}_n^T y - \theta\|_{\text{F}}^2,$$

and thus the M-step (3.41) can be solved separately for α and (θ, μ).

3.4.4 A Connection to Bipartite Graph Projection

The clustering model (3.33) involves latent variables μ that do not appear in the likelihood (3.32). This allows us to derive an efficient EM algorithm in Section 3.4.3. To better understand (3.33), we connect the bipartite graphical structure between θ and μ to a graphical structure on θ alone. Given a $\gamma = \{\gamma_{ij}\}$ that satisfies (3.34), we call it non-degenerate if $\sum_{i=1}^{n} \gamma_{ij} > 0$ for all $j \in [k]$. In other words, none of the k clusters is empty.

Proposition 3.4.1. *Let the conditional distribution of $\theta, \mu|\gamma, \sigma^2$ be specified by (3.33) with some non-degenerate γ. Then, the distribution of $\theta|\gamma, \sigma^2$ weakly converges to*

$$p(\theta|\gamma, \sigma^2) \propto \prod_{1 \le i < l \le n} \exp\left(-\frac{\lambda_{il}\|\theta_i - \theta_l\|^2}{2\sigma^2 v_0}\right) \mathbb{I}\{\mathbb{1}_n^T \theta = 0\}, \tag{3.42}$$

as $v_1 \to \infty$, where $\lambda_{il} = \sum_{j=1}^{k} \gamma_{ij}\gamma_{lj}/n_j$ with $n_j = \sum_{i=1}^{n} \gamma_{ij}$ being the size of the jth cluster.

The formula (3.42) resembles (3.3), except that λ encodes a clustering structure. By the definition of λ_{il}, if θ_i and θ_l are in different clusters, $\lambda_{il} = 0$, and otherwise, λ_{il} takes the inverse of the size of the cluster that both θ_i and θ_l belong to. The relation between (3.33) and (3.42) can be understood from the operations of graph projection and graph lift, in the sense that the weighted graph λ, with nodes θ, is a projection of γ, a bipartite graph between nodes θ and nodes μ. Conversely, γ is said to be a graph lift of λ. Observe that the clustering structure of (3.42) is combinatorial, and therefore it is much easier to work with the bipartite structure in (3.33) with latent variables.

3.4.5 A Connection to Gaussian Mixture Models

We establish a connection to Gaussian mixture models. We first give the following result.

Proposition 3.4.2. *Let the conditional distribution of $\theta, \mu|\gamma, \sigma^2$ be specified by (3.33). Then, as $v_0 \to 0$, this conditional distribution weakly converges to the distribution specified by the following*

sampling process: $\theta = \gamma\mu$ *and* $\mu|\gamma, \sigma^2$ *is sampled from*

$$p(\mu|\gamma,\sigma^2) \propto \prod_{1\le j<l\le k} \exp\left(-\frac{(n_j+n_l)\|\mu_j-\mu_l\|^2}{2\sigma^2 v_1}\right) \mathbb{I}\{\mathbb{1}_n^T\gamma\mu = 0\}, \tag{3.43}$$

where $n_j = \sum_{i=1}^n \gamma_{ij}$ *being the size of the* j*th cluster.*

With this proposition, we can see that as $v_0 \to 0$, the clustering model specified by (3.32), (3.30) and (3.33) becomes

$$y|\alpha,\mu,\gamma,\sigma^2 \sim N(\mathbb{1}_n\alpha^T + \gamma\mu, \sigma^2 I_n \otimes I_d), \tag{3.44}$$

with $\alpha|\sigma^2$ distributed by (3.30) and $\mu|\gamma,\sigma^2$ distributed by (3.43). The likelihood function (3.44) is commonly used in Gaussian mixture models, which encodes an exact clustering structure. Therefore, with a finite v_0, the model specified by (3.32), (3.30) and (3.33) can be interpreted as a relaxed version of the Gaussian mixture models that leads to an approximate clustering structure.

3.4.6 Adaptation to the Number of Clusters

The number k in (3.33) should be understood as an upper bound of the number of clusters. Even though the EM algorithm outputs k cluster centers $\{\widehat{\mu}_1, ..., \widehat{\mu}_k\}$, these k cluster centers will be automatically grouped according to their own closedness as we vary the value of v_0. Generally speaking, for a very small v_0 (the Gaussian mixture model in Section 3.4.5, for example), $\{\widehat{\mu}_1, ..., \widehat{\mu}_k\}$ will take k vectors that are not close to each other. As we increase v_0, the clustering centers $\{\widehat{\mu}_1, ..., \widehat{\mu}_k\}$ start to merge, and eventually for a sufficiently large v_0, they will all converge to a single vector. In short, v_0 parametrizes the solution path of our clustering algorithm, and on this solution path, the *effective* number of clusters increases as the value of v_0 increases.

We illustrate this point by a simple numerical example. Let us consider the observation $y = (4, 2, -2, 4)^T \in \mathbb{R}^{4\times 1}$. We fit our clustering model with $k \in \{2, 3, 4\}$. Figure 3.1 visualizes the output of the EM algorithm $(\widehat{\mu}, \widehat{\theta})$ as v_0 varies. It is clear that the solution path always starts at $\{\widehat{\mu}_1, ..., \widehat{\mu}_k\}$ of different values. Then, as v_0 increases, the solution path has various phase transitions

where the closest two $\widehat{\mu}_j$'s merge. In the end, for a sufficiently large v_0, the clustering centers $\{\widehat{\mu}_1, ..., \widehat{\mu}_k\}$ all merge to a common value.

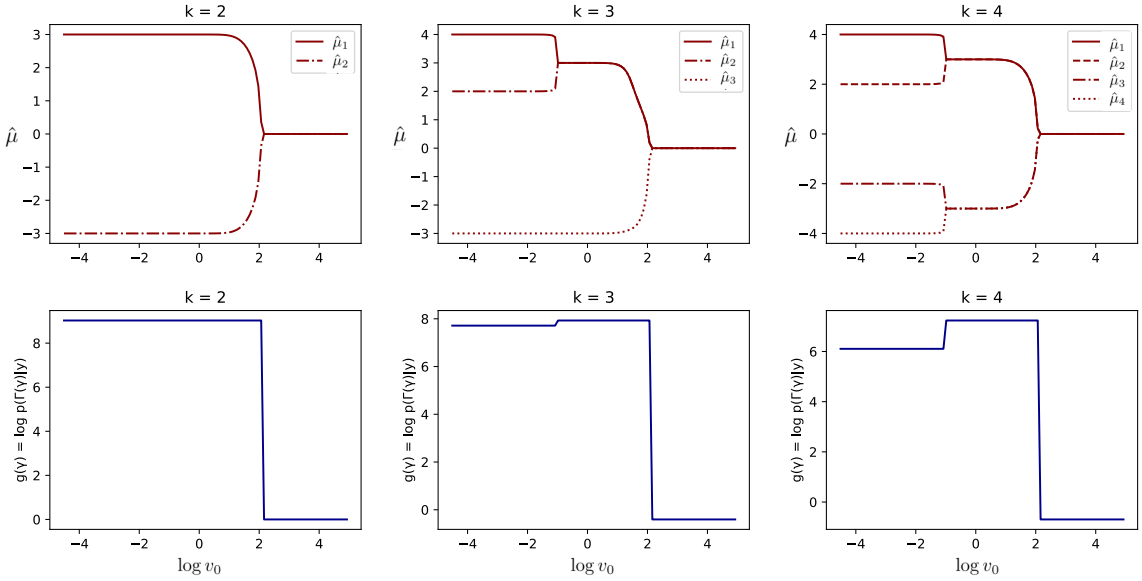


Figure 3.1: (*Top*) Solution paths of $\widehat{\mu}$ with different choices of k; (*Bottom*) Model selection scores on the solution paths.

To explain this phenomenon, it is most clear to investigate the case $k = 2$. Then, the M-step (3.41) updates μ according to

$$\begin{aligned}
\mu_1^{\text{new}} &= \underset{\mu_1}{\text{argmin}} \sum_{i=1}^{n} \left(\frac{q_{i1}}{v_0} + \frac{1 - q_{i1}}{v_1} \right) \|\theta_i - \mu_1\|^2 \\
\mu_1^{\text{new}} &= \underset{\mu_2}{\text{argmin}} \sum_{i=1}^{n} \left(\frac{q_{i2}}{v_0} + \frac{1 - q_{i2}}{v_1} \right) \|\theta_i - \mu_2\|^2.
\end{aligned}$$

Observe that both μ_1^{new} and μ_1^{new} are weighted averages of $\{\theta_1, ..., \theta_n\}$, and the only difference

between μ_1^{new} and μ_1^{new} lies in the weights. According to the E-step (3.40),

$$\begin{aligned} q_{i1} &= \frac{\exp\left(-\frac{\|\theta_i-\mu_1\|^2}{2\sigma^2\bar{v}}\right)}{\exp\left(-\frac{\|\theta_i-\mu_1\|^2}{2\sigma^2\bar{v}}\right)+\exp\left(-\frac{\|\theta_i-\mu_2\|^2}{2\sigma^2\bar{v}}\right)} \\ q_{i2} &= \frac{\exp\left(-\frac{\|\theta_i-\mu_2\|^2}{2\sigma^2\bar{v}}\right)}{\exp\left(-\frac{\|\theta_i-\mu_1\|^2}{2\sigma^2\bar{v}}\right)+\exp\left(-\frac{\|\theta_i-\mu_2\|^2}{2\sigma^2\bar{v}}\right)}, \end{aligned}$$

and we recall that $\bar{v}^{-1} = v_0^{-1} - v_1^{-1}$. Therefore, as $v_0 \to \infty$, $q_{i1} \to 1/2$ and $q_{i2} \to 1/2$, which results in the phenomenon that μ_1^{new} and μ_2^{new} merge to the same value. The same reasoning also applies to $k \geq 2$.

3.4.7 Model Selection

In this section, we discuss how to select a clustering structure from the solution path of the EM algorithm. According to the discussion in Section 3.4.6, we should understand k as an upper bound of the number of clusters, and the estimator of the number of clusters will be part of the output of the model selection procedure. The general recipe of our method follows the framework discussed in Section 3.3.3, but some nontrivial twist is required in the clustering problem. To make the presentation clear, the model selection procedure will be introduced in two parts. We will first propose our model selection score, and then we will describe a method that extracts a clustering structure from the output of the EM algorithm.

The model selection score. For any $\gamma \in \{0,1\}^{n\times k}$ that satisfies (3.34), we can calculate the posterior probability $p(\gamma|y)$ with $v_0 = 0$. This can be done by the connection to Gaussian mixture models discussed in Section 3.4.5. To be specific, the calculation follows the formula

$$p(\gamma|y) = \int\int\int p(y|\alpha,\mu,\gamma,\sigma^2)p(\alpha|\sigma^2)p(\mu|\gamma,\sigma^2)p(\sigma^2)p(\gamma)d\alpha d\mu d\sigma^2,$$

where $p(y|\alpha,\mu,\gamma,\sigma^2)$, $p(\alpha|\sigma^2)$, $p(\mu|\gamma,\sigma^2)$, $p(\sigma^2)$, and $p(\gamma)$ are specified by (3.32), (3.30), (3.43), (3.6) and (3.35). A standard calculation gives the formula

$$p(\gamma|y) \propto \left(\frac{\nu}{\nu+n} \times \frac{\det_{\gamma^T\mathbb{1}_n}(\bar{L}_\gamma)}{\det_{\gamma^T\mathbb{1}_n}(\bar{L}_\gamma+\gamma^T\gamma)}\right)^{d/2} \times \left[\frac{\nu n}{\nu+n}\|n^{-1}\mathbb{1}_n y\|^2 + \mathsf{Tr}\left((y-n^{-1}\mathbb{1}_n\mathbb{1}_n^T y)^T(I_n-\gamma(\bar{L}_\gamma+\gamma^T\gamma)\gamma^T)(y-n^{-1}\mathbb{1}_n\mathbb{1}_n^T y)\right)\right]^{-\frac{nd+a}{2}}, \tag{3.45}$$

where $\bar{L}_\gamma = (\mathbb{1}_{k\times n}\gamma + \gamma^T\mathbb{1}_{n\times k} - 2\gamma^T\gamma)/v_1$ is the graph Laplacian of the weighted adjacency matrix, which satisfies

$$\mathsf{Tr}(\mu^T\bar{L}_\gamma\mu) = \sum_{1\le j<l\le k}\frac{n_j+n_l}{v_1}\|\mu_j-\mu_l\|^2.$$

Recall that $n_j = \sum_{i=1}^n \gamma_{ij}$ is the size of the jth cluster.

However, the goal of the model selection is to select a clustering structure, and it is possible that different γ's may correspond to the same clustering structure due to label permutation. To overcome this issue, we need to sum over all equivalent γ's. Given a $\gamma \in \{0,1\}^{n\times k}$ that satisfies (3.34), define a symmetric membership matrix $\Gamma(\gamma) \in \{0,1\}^{n\times n}$ by

$$\Gamma_{il}(\gamma) = \mathbb{I}\left\{\sum_{j=1}^k \gamma_{ij}\gamma_{lj} = 1\right\}.$$

In other words, $\Gamma_{il}(\gamma) = 1$ if and only if i and l are in the same cluster. It is easy to see that every clustering structure can be uniquely represented by a symmetric membership matrix. We define the posterior probability of a clustering structure Γ by

$$p(\Gamma|y) = \sum_{\gamma\in\{\gamma:\Gamma(\gamma)=\Gamma\}} p(\gamma|y).$$

The explicit calculation of the above summation is not necessary. A shortcut can be derived from the

fact that $p(\gamma|y) = p(\gamma'|y)$ if $\Gamma(\gamma) = \Gamma(\gamma')$. This immediately implies that $p(\Gamma|y) = |\Gamma| p(\gamma|y)$ for any γ that satisfies $\Gamma(\gamma) = \Gamma$. Suppose Γ encodes a clustering structure with $\widetilde{k}$ nonempty clusters, and then we have $|\Gamma| = \binom{k}{\widetilde{k}} \widetilde{k}!$. This leads to the model selection score

$$g(\gamma) = \log p(\Gamma(\gamma)|y) = \log p(\gamma|y) + \log \left[\binom{k}{\widetilde{k}} \widetilde{k}! \right], \tag{3.46}$$

for any $\gamma \in \{0,1\}^{n \times k}$ that satisfies (3.34), and $\widetilde{k}$ above is calculated by $\widetilde{k} = \sum_{j=1}^{k} \max_{1 \le i \le n} \gamma_{ij}$, the effective number of clusters.

Extraction of clustering structures from the EM algorithm. Let $\widehat{\mu}$ and $\widehat{q}$ be outputs of the EM algorithm, and we discuss how to obtain $\widehat{\gamma}$ that encodes a meaningful clustering structure to be evaluated by the model selection score (3.46). It is very tempting to directly threshold $\widehat{q}$ as is done in Section 3.3.3. However, as has been discussed in Section 3.4.6, the solution paths of $\{\mu_1, ..., \mu_k\}$ merge at some values of v_0. Therefore, we should treat the clusters whose clustering centers merge together as a single cluster.

Given $\widehat{\mu}_1, ..., \widehat{\mu}_k$ output by the M-step, we first merge $\widehat{\mu}_j$ and $\widehat{\mu}_l$ whenever $\|\widehat{\mu}_j - \widehat{\mu}_l\| \le \epsilon$. The number ϵ is taken as 10^{-8}, the square root of the machine precision, in our code. This forms a partition $[k] = \cup_{l=1}^{\widehat{k}} \mathcal{G}_l$ for some $\widehat{k} \le k$. Then, by taking average within each group, we obtain a reduced collection of clustering centers $\widetilde{\mu}_1, ..., \widetilde{\mu}_{\widehat{k}}$. In other words, $\widetilde{\mu}_l = |\mathcal{G}_l|^{-1} \sum_{j \in \mathcal{G}_l} \widehat{\mu}_j$.

The $\widehat{q} \in [0,1]^{n \times k}$ output by the E-step should also be reduced to $\widetilde{q} \in [0,1]^{n \times \widehat{k}}$ as well. Note that $\widehat{q}_{ij}$ is the estimated posterior probability that the ith node belongs to the jth cluster. This means that $\widetilde{q}_{il}$ is the estimated posterior probability that the ith node belongs to the lth reduced cluster. An explicit formua is given by $\widetilde{q}_{il} = \sum_{j \in \mathcal{G}_l} \widehat{q}_{ij}$.

With the reduced posterior probability $\widetilde{q}$, we simply apply thresholding to obtain $\widehat{\gamma}$. We have $(\widehat{\gamma}_{i1}, ..., \widehat{\gamma}_{ik}) = e_j$ if $j = \operatorname{argmax}_{1 \le l \le \widehat{k}} \widetilde{q}_{il}$. Recall that e_j is a vector with 1 on the jth entry and 0 elsewhere. Note that according to this construction, we always have $\widehat{\gamma}_{ij} = 0$ whenever $j > \widehat{k}$. This does not matter, because the model selection score (3.46) does not depend on the clustering labels. Finally, the $\widehat{\gamma}$ constructed according to the above procedure will be evaluated by $g(\widehat{\gamma})$ defined by

(3.46).

In the toy example with four data points $y = (4, 2, -2, 4)^T$, the model selection score is computed along the solution path. According to Figure 3.1, the model selection procedure suggests that a clustering structure with two clusters $\{4, 2\}$ and $\{-2, -4\}$ is the most plausible one. We also note that the curve of $g(\gamma)$ has sharp phase transitions whenever the solution paths μ merge.

3.5 Extensions with Graph Algebra

In many applications, it is useful to have a model that imposes both row and column structures on a high-dimensional matrix $\theta \in \mathbb{R}^{p_1 \times p_2}$. We list some important examples below.

1. *Biclustering.* In applications such as gene expression data analysis, one needs to cluster both samples and features. This task imposes a clustering structure for both rows and columns of the data matrix [71, 29].

2. *Block sparsity.* In problems such as planted clique detection [46] and submatrix localization [67], the matrix can be viewed as the sum of a noise background plus a submatrix of signals with unknown locations. Equivalently, it can be modeled by simultaneous row and column sparsity [102].

3. *Sparse clustering.* Suppose the data matrix exhibits a clustering structure for its rows and a sparsity structure for its columns, then we have a sparse clustering problem [166]. For this task, we need to select nonzero column features in order to accurately cluster the rows.

For the problems listed above, the row and column structures can be modeled by graphs γ_1 and γ_2. Then, the structure of the matrix θ is induced by a notion of graph product of γ_1 and γ_2. In this section, we introduce tools from graph algebra including Cartesian product and Kronecker product to build complex structure from simple components.

We first introduce the likelihood of the problem. To cope with many useful models, we assume that the observation can be organized as a matrix $y \in \mathbb{R}^{n_1 \times n_2}$. Then, the specific setting of a certain

problem can be encoded by design matrices $X_1 \in \mathbb{R}^{n_1 \times p_1}$ and $X_2 \in \mathbb{R}^{n_2 \times p_2}$. The likelihood is defined by

$$y|\alpha, \theta, \sigma^2 \sim N(X_1(\alpha w + \theta)X_2^T, \sigma^2 I_{n_1} \otimes I_{n_2}). \tag{3.47}$$

The matrix $w \in \mathbb{R}^{p_1 \times p_2}$ is assumed to have rank one, and can be decomposed as $w = w_1 w_2^T$ for some $w_1 \in \mathbb{R}^{p_1}$ and $w_2 \in \mathbb{R}^{p_2}$. The prior distribution of the scalar is simply given by

$$\alpha|\sigma^2 \sim N(0, \sigma^2/\nu). \tag{3.48}$$

We then need to build prior distributions of θ that is supported on $\Theta_w = \{\theta \in \mathbb{R}^{p_1 \times p_2} : \mathsf{Tr}(w\theta^T) = 0\}$ using Cartesian and Kronecker products.

3.5.1 Cartesian Product

We start with the definition of the Cartesian product of two graphs.

Definition 3.5.1. *Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, their Cartesian product $G = G_1 \square G_2$ is defined with the vertex set $V_1 \times V_2$. Its edge set contains $((x_1, x_2), (y_1, y_2))$ if and only if $x_1 = y_1$ and $(x_2, y_2) \in E_2$ or $(x_1, y_1) \in E_1$ and $x_2 = y_2$.*

According to the definition, it can be checked that for two graphs of sizes p_1 and p_2, the adjacency matrix, the Laplacian and the incidence matrix of the Cartesian product enjoy the relations

$$\begin{aligned} A_{1\square 2} &= A_2 \otimes I_{p_1} + I_{p_2} \otimes A_1, \\ L_{1\square 2} &= L_2 \otimes I_{p_1} + I_{p_2} \otimes L_1, \\ D_{1\square 2} &= [D_2 \otimes I_{p_1}; I_{p_2} \otimes D_1]. \end{aligned}$$

Given graphs γ_1 and γ_2 that encode row and column structures of θ, we introduce the following

prior distribution

$$p(\theta|\gamma_1,\gamma_2,\sigma^2) \propto \prod_{(i,j)\in E_1} \exp\left(-\frac{\|\theta_{i*}-\theta_{j*}\|^2}{2\sigma^2[v_0\gamma_{1,ij}+v_1(1-\gamma_{1,ij})]}\right) \times \prod_{(k,l)\in E_2} \exp\left(-\frac{\|\theta_{*k}-\theta_{*l}\|^2}{2\sigma^2[v_0\gamma_{2,kl}+v_1(1-\gamma_{2,kl})]}\right) \mathbb{I}\{\theta\in\Theta_w\}. \tag{3.49}$$

Here, E_1 and E_2 are the base graphs of the row and column structures. According to its form, the prior distribution (3.49) models both pairwise relations of rows and those of columns based on γ_1 and γ_2, respectively. To better understand (3.49), we can write it in the following equivalent form,

$$p(\theta|\gamma_1,\gamma_2,\sigma^2) \propto \exp\left(-\frac{1}{2\sigma^2}\mathsf{vec}(\theta)^T\left(L_{\gamma_2}\otimes I_{p_1}+I_{p_2}\otimes L_{\gamma_1}\right)\mathsf{vec}(\theta)\right)\mathbb{I}\{\theta\in\Theta_w\}, \tag{3.50}$$

where $L_{\gamma_1}\in\mathbb{R}^{p_1\times p_1}$ and $L_{\gamma_2}\in\mathbb{R}^{p_2\times p_2}$ are Laplacian matrices of the weighted graphs $\{v_0\gamma_{1,ij}+v_1(1-\gamma_{1,ij})\}$ and $\{v_0\gamma_{2,kl}+v_1(1-\gamma_{2,kl})\}$, respectively. Therefore, by Definition 3.5.1, $p(\theta|\gamma_1,\gamma_2,\sigma^2)$ is a spike-and-slab Laplacian prior $p(\theta|\gamma,\sigma^2)$ defined in (3.3) with $\gamma=\gamma_1\square\gamma_2$, and the well-definedness is guaranteed by Proposition 3.2.1.

To complete the Bayesian model, the distribution of $(\gamma_1,\gamma_2,\sigma^2)$ are specified by

$$\gamma_1,\gamma_2|\eta_1,\eta_2 \sim \prod_{(i,j)\in E_1}\eta_1^{\gamma_{1,ij}}(1-\eta_1)^{1-\gamma_{1,ij}}\prod_{(i,j)\in E_2}\eta_2^{\gamma_{2,kl}}(1-\eta_2)^{1-\gamma_{2,kl}}, \tag{3.51}$$

$$\eta_1,\eta_2 \sim \text{Beta}(A_1,B_1)\bigotimes\text{Beta}(A_2,B_2), \tag{3.52}$$

$$\sigma^2 \sim \text{InvGamma}(a/2,b/2). \tag{3.53}$$

We remark that it is possible to constrain γ_1 and γ_2 in some subsets Γ_1 and Γ_2 like (3.4). This extra twist is useful for a biclustering model that will be discussed in Section 3.5.3.

Note that in general the base graph $G=G_1\square G_2$ is not a tree, and the derivation of the EM algorithm follows a similar argument in Section 3.3.2. Using the same argument in (3.19), we lower

bound $\log \sum_{T \in \text{spt}(G)} \sum_{e \in T} [v_0^{-1}\gamma_e + v_1^{-1}(1-\gamma_e)]$ by

$$\sum_{e \in E_1 \square E_2} r_e \log[v_0^{-1}\gamma_e + v_1^{-1}(1-\gamma_e)] + \log|\text{spt}(G)|. \tag{3.54}$$

Since

$$E_1 \square E_2 = \{((i,k),(j,k)) : (i,j) \in E_1, k \in V_2\} \cup \{((i,k),(i,l)) : i \in V_1, (k,l) \in E_2\},$$

and $\gamma = \gamma_1 \square \gamma_2$, we can write (3.54) as

$$\begin{aligned}
&\sum_{(i,j)\in E_1} \sum_{k=1}^{p_2} r_{(i,k),(j,k)} \log[v_0^{-1}\gamma_{1,ij} + v_1^{-1}(1-\gamma_{1,ij})] \\
&+ \sum_{(k,l)\in E_2} \sum_{i=1}^{p_1} r_{(i,k),(i,l)} \log[v_0^{-1}\gamma_{2,kl} + v_1^{-1}(1-\gamma_{2,kl})] \\
= &\sum_{(i,j)\in E_1} r_{1,ij} \log[v_0^{-1}\gamma_{1,ij} + v_1^{-1}(1-\gamma_{1,ij})] \\
&+ \sum_{(k,l)\in E_2} r_{2,kl} \log[v_0^{-1}\gamma_{2,kl} + v_1^{-1}(1-\gamma_{2,kl})],
\end{aligned}$$

where

$$r_{1,ij} = \sum_{k=1}^{p_2} r_{(i,k),(j,k)} = \frac{1}{|\text{spt}(G)|} \sum_{k=1}^{p_2} \sum_{T \in \text{spt}(G)} \mathbb{I}\{((i,k),(j,k)) \in T\},$$

and $r_{2,kl}$ is similarly defined.

Using the lower bound derived above, it is direct to derive the an EM algorithm, which consists

of the following iterations,

$$q_{1,ij}^{\text{new}} = \frac{\eta_1 v_0^{-\frac{r_{1,ij}}{2}} e^{-\frac{1}{2\sigma^2 v_0}\|\theta_{i*}-\theta_{j*}\|^2}}{\eta_1 v_0^{-\frac{r_{1,ij}}{2}} e^{-\frac{1}{2\sigma^2 v_0}\|\theta_{i*}-\theta_{j*}\|^2} + (1-\eta_1) v_1^{-\frac{r_{1,ij}}{2}} e^{-\frac{1}{2\sigma^2 v_1}\|\theta_{i*}-\theta_{j*}\|^2}}, \quad (3.55)$$

$$q_{2,kl}^{\text{new}} = \frac{\eta_2 v_0^{-\frac{r_{2,kl}}{2}} e^{-\frac{1}{2\sigma^2 v_0}\|\theta_{*k}-\theta_{*l}\|^2}}{\eta_2 v_0^{-\frac{r_{2,kl}}{2}} e^{-\frac{1}{2\sigma^2 v_0}\|\theta_{*k}-\theta_{*l}\|^2} + (1-\eta_2) v_1^{-\frac{r_{2,kl}}{2}} e^{-\frac{1}{2\sigma^2 v_1}\|\theta_{*k}-\theta_{*l}\|^2}}, \quad (3.56)$$

$$(\alpha^{\text{new}}, \theta^{\text{new}}) = \underset{\alpha,\theta\in\Theta_w}{\operatorname{argmin}} F(\alpha, \theta; q_1^{\text{new}}, q_2^{\text{new}}), \quad (3.57)$$

$$(\sigma^2)^{\text{new}} = \frac{F(\alpha^{\text{new}}, \theta^{\text{new}}; q_1^{\text{new}}, q_2^{\text{new}}) + b}{n_1 n_2 + p_1 p_2 + a + 2},$$

$$\eta_1^{\text{new}} = \frac{A_1 - 1 + \sum_{(i,j)\in E_1} q_{1,ij}^{\text{new}}}{A_1 + B_1 - 2 + m_1},$$

$$\eta_2^{\text{new}} = \frac{A_2 - 1 + \sum_{(k,l)\in E_2} q_{2,ij}^{\text{new}}}{A_2 + B_2 - 2 + m_2}. \quad (3.58)$$

The definition of the function $F(\alpha, \theta; q_1, q_2)$ is given by

$$F(\alpha, \theta; q_1, q_2) = \|y - X_1(\alpha w + \theta)X_2^T\|_{\text{F}}^2 + \nu\alpha^2 + \mathsf{vec}(\theta)^T \left(L_{q_2} \otimes I_{p_1} + I_{p_2} \otimes L_{q_1}\right) \mathsf{vec}(\theta)$$

Though the E-steps (3.55) and (3.56) are straightforward, the M-step (3.57) is a quadratic programming of dimension $p_1 p_2$, which may become the computational bottleneck of the EM algorithm when the size of the problem is large. We will introduce a Dykstra-like algorithm to solve (3.57) in Appendix 3.8.5.

The Cartesian product model is useful for simultaneous learning the row structure γ_1 and the column structure and γ_2 of the coefficient matrix θ. Note that when $X_1 = X$, $X_2 = I_d$, and $E_2 = \varnothing$, the Cartesian product model becomes the multivariate regression model described in Section 3.4.1. In this case, the model only regularizes the row structure of θ. Another equally interesting example is obtained when $X_1 = X$, $X_2 = I_d$, and $E_1 = \varnothing$. In this case, the model only regularizes the column structure of θ, and can be interpreted as multitask learning with task clustering.

3.5.2 Kronecker Product

The Kronecker product of two graphs is defined below.

Definition 3.5.2. *Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, their Kronecker product $G = G_1 \otimes G_2$ is defined with the vertex set $V_1 \times V_2$. Its edge set contains $((x_1, x_2), (y_1, y_2))$ if and only if $(x_1, y_1) \in E_1$ and $(x_2, y_2) \in E_2$.*

It is not hard to see that the adjacency matrix of two graphs has the formula $A_{1\otimes 2} = A_1 \otimes A_2$, which gives the name of Definition 3.5.2. The prior distribution of θ given row and column graphs γ_1 and γ_2 that we discuss in this subsection is

$$p(\theta|\gamma_1, \gamma_2, \sigma^2) \propto \prod_{(i,j)\in E_1} \prod_{(k,l)\in E_2} \exp\left(-\frac{(\theta_{ik} - \theta_{jl})^2}{2\sigma^2[v_0\gamma_{1,ij}\gamma_{2,kl} + v_1(1 - \gamma_{1,ij}\gamma_{2,kl})]}\right) \mathbb{I}\{\theta \in \Theta_w\}. \tag{3.59}$$

Again, E_1 and E_2 are the base graphs of the row and column structures. According to its form, the prior imposes a nearly block structure on θ based on the graphs γ_1 and γ_2. Moreover, $p(\theta|\gamma_1, \gamma_2, \sigma^2)$ can be viewed as a spike-and-slab Laplacian prior $p(\theta|\gamma, \sigma^2)$ defined in (3.3) with $\gamma = \gamma_1 \otimes \gamma_2$. The distribution of $(\gamma_1, \gamma_2, \sigma^2)$ follows the same specification in (3.51)-(3.53).

To derive an EM algorithm, we follow the strategy described in Section 3.3.2 and the lower bound $\log \sum_{T\in \text{spt}(G)} \sum_{e\in T}[v_0^{-1}\gamma_e + v_1^{-1}(1 - \gamma_e)]$ by

$$\sum_{(i,j)\in E_1} \sum_{(k,l)\in E_2} r_{(i,k),(j,l)} \log[v_0^{-1}\gamma_{1,ij}\gamma_{2,kl} + v_1^{-1}(1 - \gamma_{1,ij}\gamma_{2,kl})]. \tag{3.60}$$

Unlike the Cartesian product, the Kronecker product structure has a lower bound (3.60) that is not separable with respect to γ_1 and γ_2. This makes the E-step combinatorial, and does not apply to a large-scale problem. To alleviate this computational barrier, we consider a variational EM algorithm that finds the best posterior distribution of γ_1, γ_2 that can be factorized. In other words, instead of maximizing over all possible distribution q, we maximize over the mean-filed class $q \in \mathcal{Q}$, with

$\mathcal{Q} = \{q(\gamma_1, \gamma_2) = q_1(\gamma_1)q_2(\gamma_2) : q_1, q_2\}$. Then, the objective becomes

$$\max_{q_1,q_2} \max_{\alpha,\theta\in\Theta_2,\delta,\eta,\sigma^2} \sum_{\gamma_1,\gamma_2} q_1(\gamma_1)q_2(\gamma_2) \log \frac{\widetilde{p}(y, \alpha, \theta, \delta, \gamma_1, \gamma_2, \eta, \sigma^2)}{q_1(\gamma_1)q_2(\gamma_2)},$$

where $\widetilde{p}(y, \alpha, \theta, \delta, \gamma_1, \gamma_2, \eta, \sigma^2)$ is obtained by replacing $p(\theta|\gamma_1, \gamma_2, \sigma^2)$ with $\widetilde{p}(\theta|\gamma_1, \gamma_2, \sigma^2)$ in the joint distribution $p(y, \alpha, \theta, \delta, \gamma_1, \gamma_2, \eta, \sigma^2)$. Here, $\log \widetilde{p}(\theta|\gamma_1, \gamma_2, \sigma^2)$ is a lower bound for $\log p(\theta|\gamma_1, \gamma_2, \sigma^2)$ with (3.60). The E-step of the variational EM is

$$\begin{aligned} q_1^{\text{new}}(\gamma_1) &\propto \exp\left(\sum_{\gamma_2} q_2(\gamma_2) \log \widetilde{p}(y, \alpha, \theta, \delta, \gamma_1, \gamma_2, \eta, \sigma^2)\right), \\ q_2^{\text{new}}(\gamma_2) &\propto \exp\left(\sum_{\gamma_1} q_1^{\text{new}}(\gamma_1) \log \widetilde{p}(y, \alpha, \theta, \delta, \gamma_1, \gamma_2, \eta, \sigma^2)\right). \end{aligned}$$

After some simplification, we have

$$q_{1,ij}^{\text{new}} = \frac{1}{1 + \dfrac{(1-\eta_1)\prod_{(k,l)\in E_2}\left(v_1^{-\frac{r_{(i,k),(j,l)}}{2}} e^{-\frac{1}{2\sigma^2 v_1}(\theta_{ik}-\theta_{jl})^2}\right)^{q_{2,kl}}}{\eta_1 \prod_{(k,l)\in E_2}\left(v_0^{-\frac{r_{(i,k),(j,l)}}{2}} e^{-\frac{1}{2\sigma^2 v_0}(\theta_{ik}-\theta_{jl})^2}\right)^{q_{2,kl}}}}, \tag{3.61}$$

$$q_{2,kl}^{\text{new}} = \frac{1}{1 + \dfrac{(1-\eta_2)\prod_{(i,j)\in E_1}\left(v_1^{-\frac{r_{(i,k),(j,l)}}{2}} e^{-\frac{1}{2\sigma^2 v_1}(\theta_{ik}-\theta_{jl})^2}\right)^{q_{1,kl}^{\text{new}}}}{\eta_2 \prod_{(i,j)\in E_1}\left(v_0^{-\frac{r_{(i,k),(j,l)}}{2}} e^{-\frac{1}{2\sigma^2 v_0}(\theta_{ik}-\theta_{jl})^2}\right)^{q_{1,kl}^{\text{new}}}}}. \tag{3.62}$$

The M-step can be derived in a standard way, and it has the same updates as in (3.57)-(3.58), with a new definition of $F(\alpha, \theta; q_1, q_2)$ given by

$$\begin{aligned} F(\alpha, \theta; q_1, q_2) &= \|y - X_1(\alpha w + \theta)X_2\|^2 + \nu\alpha^2 \\ &\quad + \sum_{(i,j)\in E_1} \sum_{(k,l)\in E_2} \left(\frac{q_{1,ij}q_{2,kl}}{v_0} + \frac{1 - q_{1,ij}q_{2,kl}}{v_1}\right)(\theta_{ik} - \theta_{jl})^2. \end{aligned}$$

3.5.3 *Applications in Biclustering*

When both row and column graphs encode clustering structures discussed in Section 3.4.2, we have the biclustering model. In this section, we discuss both biclustering models induced by Kronecker and Cartesian products. We start with a special form of the likelihood (3.47), which is given by

$$y|\alpha, \theta, \sigma^2 \sim N(\alpha \mathbb{1}_{n_1} \mathbb{1}_{n_2}^T + \theta, \sigma^2 I_{n_1} \otimes I_{n_2}),$$

and the prior distribution on α is given by (3.48). The prior distribution on θ will be discussed in two cases.

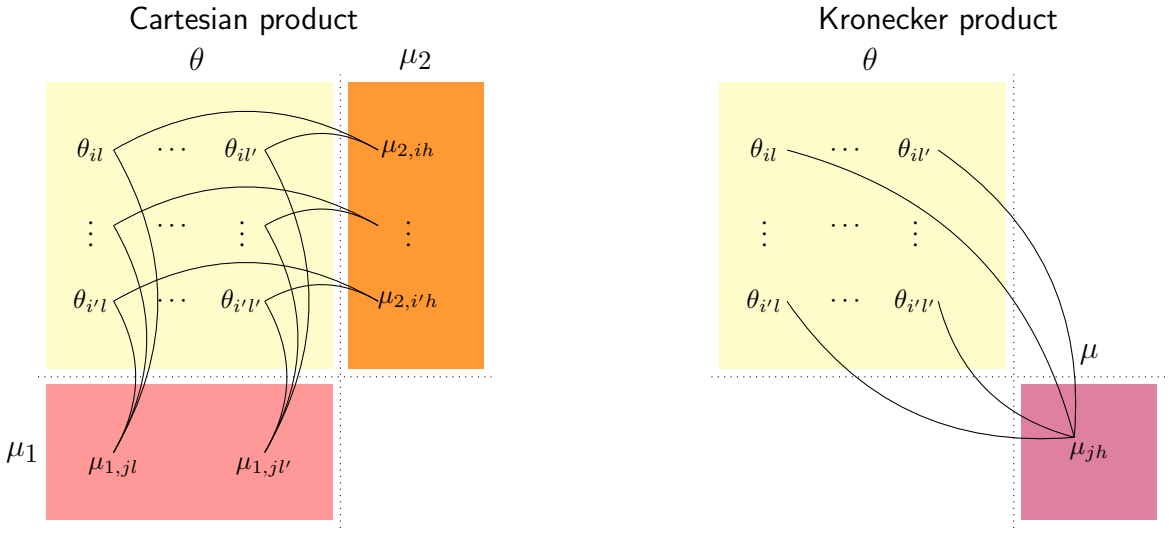


Figure 3.2: Structure diagrams for the two biclustering methods. The Cartesian product biclustering model *(Left)* and the Kronecker product biclustering model *(Right)* have different latent variables and base graphs. While the Cartesian product models the row and column clustering structures by separate latent variable matrices $\mu_1 \in \mathbb{R}^{k_1 \times n_2}$ and $\mu_2 \in \mathbb{R}^{n_1 \times k_2}$, the Kronecker product directly models the checkerboard structure by a single latent matrix $\mu \in \mathbb{R}^{k_1 \times k_2}$.

Cartesian product biclustering model. Let $k_1 \in [n_1]$ and $k_2 \in [n_2]$ be upper bounds of the numbers of row and column clusters, respectively. We introduce two latent matrices $\mu_1 \in \mathbb{R}^{k_1 \times n_2}$ and $\mu_2 \in \mathbb{R}^{n_1 \times k_2}$ that serve as row and column clustering centers. The prior distribution is then

specified by

$$p(\theta, \mu_1, \mu_2 | \gamma_1, \gamma_2, \sigma^2) \propto \prod_{i=1}^{n_1} \prod_{j=1}^{k_1} \exp\left(-\frac{\|\theta_{i*} - \mu_{1,j*}\|^2}{2\sigma^2[v_0\gamma_{1,ij} + v_1(1-\gamma_{1,ij})]} \right) \times \prod_{l=1}^{n_2} \prod_{h=1}^{k_2} \exp\left(-\frac{\|\theta_{*l} - \mu_{2,*h}\|^2}{2\sigma^2[v_0\gamma_{2,lh} + v_1(1-\gamma_{2,lh})]} \right) \mathbb{I}\{\mathbb{1}_{n_1}^T \theta \mathbb{1}_{n_2} = 0\},$$

which can be regarded as an extension of (3.33) in the form of (3.49). The prior distributions on γ_1 and γ_2 are independently specified by (3.35) with (k, n) replaced by (k_1, n_1) and (k_2, n_2). Finally, σ^2 follows the inverse Gamma prior (3.6).

We follow the framework of Section 3.3.2. The derivation of the EM algorithm requires lower bounding $\log \sum_{T \in \text{spt}(G)} \sum_{e \in T} [v_0^{-1}\gamma_e + v_1^{-1}(1-\gamma_e)]$. Using the same argument in Section 3.5.1, we have the following lower bound

$$\sum_{i=1}^{n_1} \sum_{j=1}^{k_1} r_{1,ij} \log[v_0^{-1}\gamma_{1,ij} + v_1^{-1}(1-\gamma_{1,ij})] + \sum_{l=1}^{n_2} \sum_{h=1}^{k_2} r_{2,lh} \log[v_0^{-1}\gamma_{2,lh} + v_1^{-1}(1-\gamma_{2,lh})]. \quad (3.63)$$

By the symmetry of the complete bipartite graph, $r_{1,ij}$ is a constant that does not depend on (i, j). Then use the same argument in (3.37)-(3.38), and we obtain the fact that

$$\sum_{i=1}^{n_1} \sum_{j=1}^{k_1} r_{1,ij} \log[v_0^{-1}\gamma_{1,ij} + v_1^{-1}(1-\gamma_{1,ij})]$$

is independent of $\{\gamma_{1,ij}\}$, and the same conclusion also applies to the second term of (3.63).

Since the lower bound (3.63) does not dependent on γ_1, γ_2, the determinant factor in the density function $p(\theta, \mu_1, \mu_2 | \gamma_1, \gamma_2, \sigma^2)$ does not play any role in the derivation of the EM algorithm. With

some standard calculations, the E-step is given by

$$\begin{aligned} q_{1,ij}^{\text{new}} &= \frac{\exp\left(-\frac{\|\theta_{i*}-\mu_{1,j*}\|^2}{2\sigma^2\bar{v}}\right)}{\sum_{u=1}^{k_1}\exp\left(-\frac{\|\theta_{i*}-\mu_{1,u*}\|^2}{2\sigma^2\bar{v}}\right)}, \\ q_{1,lh}^{\text{new}} &= \frac{\exp\left(-\frac{\|\theta_{*l}-\mu_{2,*h}\|^2}{2\sigma^2\bar{v}}\right)}{\sum_{v=1}^{k_2}\exp\left(-\frac{\|\theta_{*l}-\mu_{2,*v}\|^2}{2\sigma^2\bar{v}}\right)}, \end{aligned}$$

where $\bar{v}^{-1} = v_0^{-1} - v_1^{-1}$. The M-step is given by

$$\begin{aligned} (\alpha^{\text{new}}, \theta^{\text{new}}, \mu_1^{\text{new}}, \mu_2^{\text{new}}) &= \underset{\alpha, \mathbb{1}_{n_1}^T\theta\mathbb{1}_{n_2}=0, \mu_1, \mu_2}{\operatorname{argmin}} F(\alpha, \theta, \mu_1, \mu_2; q_1^{\text{new}}, q_2^{\text{new}}), \\ (\sigma^2)^{\text{new}} &= \frac{F(\alpha^{\text{new}}, \theta^{\text{new}}, \mu_1^{\text{new}}, \mu_2^{\text{new}}; q_1^{\text{new}}, q_2^{\text{new}}) + b}{2n_1n_2 + n_1k_2 + n_2k_1 + a + 2}, \end{aligned}$$

where

$$\begin{aligned} F(\alpha, \theta, \mu_1, \mu_2; q_1, q_2) &= \|y - \alpha\mathbb{1}_{n_1}\mathbb{1}_{n_2}^T - \theta\|_{\text{F}}^2 + \nu\|\alpha\|^2 \\ &\quad + \sum_{i=1}^{n_1}\sum_{j=1}^{k_1}\left(\frac{q_{1,ij}}{v_0} + \frac{1-q_{1,ij}}{v_1}\right)\|\theta_{i*} - \mu_{1,j*}\|^2 \\ &\quad + \sum_{l=1}^{n_2}\sum_{h=1}^{k_2}\left(\frac{q_{2,lh}}{v_0} + \frac{1-q_{2,lh}}{v_1}\right)\|\theta_{*l} - \mu_{2,*h}\|^2. \end{aligned}$$

Kronecker product biclustering model. For the Kronecker product structure, we introduce a latent matrix $\mu \in \mathbb{R}^{k_1 \times k_2}$. Since the biclustering model implies a block-wise constant structure for θ. Each entry of μ serves as a center for a block of the matrix θ. The prior distribution is defined by

$$p(\theta, \mu|\gamma_1, \gamma_2, \sigma^2) \propto \prod_{i,j,l,h}\exp\left(-\frac{(\theta_{il} - \mu_{jh})^2}{2\sigma^2[v_0\gamma_{1,ij}\gamma_{2,lh} + v_1(1 - \gamma_{1,ij}\gamma_{2,lh})]}\right)\mathbb{I}\{\mathbb{1}_{n_1}^T\theta\mathbb{1}_{n_2} = 0\}.$$

The prior distribution is another extension of (3.33), and it is in a similar form of (3.59). To finish the Bayesian model specification, we consider the same priors for $\gamma_1, \gamma_2, \sigma^2$ as in the Cartesian product case.

Recall that the lower bound of $\log \sum_{T \in \text{spt}(G)} \sum_{e \in T} [v_0^{-1}\gamma_e + v_1^{-1}(1-\gamma_e)]$ is given by (3.60) for a general Kronecker product structure. In the current setting, a similar argument gives the lower bound

$$\sum_{i=1}^{n_1}\sum_{j=1}^{k_1}\sum_{l=1}^{n_2}\sum_{h=1}^{k_2} r_{(i,l),(j,h)} \log[v_0^{-1}\gamma_{1,ij}\gamma_{2,lh} + v_1^{-1}(1-\gamma_{1,ij}\gamma_{2,lh})].$$

Since $r_{(i,l),(j,h)}$ is independent of $(i,l),(j,h)$ by the symmetry of the complete bipartite graph, the above lower bound can be written as

$$\begin{aligned} & r\sum_{i=1}^{n_1}\sum_{j=1}^{k_1}\sum_{l=1}^{n_2}\sum_{h=1}^{k_2} \log[v_0^{-1}\gamma_{1,ij}\gamma_{2,lh} + v_1^{-1}(1-\gamma_{1,ij}\gamma_{2,lh})] \\ = \;& r\log(v_0^{-1})\sum_{i=1}^{n_1}\sum_{j=1}^{k_1}\sum_{l=1}^{n_2}\sum_{h=1}^{k_2}\gamma_{1,ij}\gamma_{2,lh} + r\log(v_1^{-1})\sum_{i=1}^{n_1}\sum_{j=1}^{k_1}\sum_{l=1}^{n_2}\sum_{h=1}^{k_2}(1-\gamma_{1,ij}\gamma_{2,lh}) \\ = \;& rn_1n_2\log(v_0^{-1}) + rn_1n_2(k_1k_2-1)\log(v_1^{-1}), \end{aligned} \tag{3.64}$$

which is independent of γ_1, γ_2. The inequality (3.64) is because both γ_1 and γ_2 satisfy (3.34).

Again, the determinant factor in the density function $p(\theta, \mu|\gamma_1, \gamma_2, \sigma^2)$ does not play any role in the derivation of the EM algorithm, because the lower bound (3.64) does not depend on (γ_1, γ_2). Since we are working with the Kronecker product, we will derive a variational EM algorithm with the E-step finding the posterior distribution in the mean filed class $\mathcal{Q} = \{q(\gamma_1, \gamma_2) = q_1(\gamma_1)q_2(\gamma_2) :$

$q_1, q_2\}$. By following the same argument in Section 3.5.2, we obtain the E-step as

$$\begin{aligned} q_{1,ij}^{\text{new}} &= \frac{\exp\left(-\sum_{l=1}^{n_2}\sum_{h=1}^{k_2}\frac{q_{2,lh}(\theta_{il}-\mu_{jh})^2}{2\sigma^2\bar{v}}\right)}{\sum_{u=1}^{k_1}\exp\left(-\sum_{l=1}^{n_2}\sum_{h=1}^{k_2}\frac{q_{2,lh}(\theta_{il}-\mu_{uh})^2}{2\sigma^2\bar{v}}\right)}, \\ q_{2,lh}^{\text{new}} &= \frac{\exp\left(-\sum_{i=1}^{n_1}\sum_{j=1}^{k_1}\frac{q_{1,ij}^{\text{new}}(\theta_{il}-\mu_{jh})^2}{2\sigma^2\bar{v}}\right)}{\sum_{v=1}^{k_2}\exp\left(-\sum_{i=1}^{n_1}\sum_{v=1}^{k_1}\frac{q_{1,ij}^{\text{new}}(\theta_{il}-\mu_{lv})^2}{2\sigma^2\bar{v}}\right)} \end{aligned}$$

where $\bar{v}^{-1} = v_0^{-1} - v_1^{-1}$. The M-step is given by

$$\begin{aligned} (\alpha^{\text{new}}, \theta^{\text{new}}, \mu^{\text{new}}) &= \underset{\alpha, \mathbb{1}_{n_1}^T\theta\mathbb{1}_{n_2}=0, \mu}{\text{argmin}} F(\alpha, \theta, \mu; q_1^{\text{new}}, q_2^{\text{new}}), \\ (\sigma^2)^{\text{new}} &= \frac{F(\alpha^{\text{new}}, \theta^{\text{new}}, \mu^{\text{new}}; q_1^{\text{new}}, q_2^{\text{new}}) + b}{2n_1n_2 + n_1k_2 + n_2k_1 + a + 2}, \end{aligned}$$

where

$$\begin{aligned} F(\alpha, \theta, \mu_1, \mu_2; q_1, q_2) &= \|y - \alpha\mathbb{1}_{n_1}\mathbb{1}_{n_2}^T - \theta\|_{\text{F}}^2 + \nu\|\alpha\|^2 \\ &\quad + \sum_{i=1}^{n_1}\sum_{j=1}^{k_1}\sum_{l=1}^{n_2}\sum_{h=1}^{k_2}\left(\frac{q_{1,ij}q_{2,lh}}{v_0} + \frac{1 - q_{1,ij}q_{2,lh}}{v_1}\right)(\theta_{il} - \mu_{jh})^2. \end{aligned}$$

3.6 Reduced Isotonic Regression

The models that we have discussed so far in our general framework all involve Gaussian likelihood functions and Gaussian priors. It is important to develop a natural extension of the framework to include non-Gaussian models. In this section, we discuss a reduced isotonic regression problem with a non-Gaussian prior distribution, while a full extension to non-Gaussian models will be considered as a future project.

Given a vector of observation $y \in \mathbb{R}^n$, the reduced isotonic regression seeks the best piecewise constant fit that is nondecreasing [134, 51]. It is an important model that has applications in problems

with natural monotone constraint on the signal. With the likelihood $y|\alpha, \theta, \sigma^2 \sim N(\alpha \mathbb{1}_n + \theta, \sigma^2 I_n)$, we need to specify a prior distribution on θ that induces both piecewise constant and isotonic structures. We propose the following prior distribution,

$$\theta|\gamma, \sigma^2 \sim p(\theta|\gamma, \sigma^2) \propto \prod_{i=1}^{n-1} \exp\left(-\frac{(\theta_{i+1}-\theta_i)^2}{2\sigma^2[v_0\gamma_i + v_1(1-\gamma_i)]}\right) \mathbb{I}\{\theta_i \le \theta_{i+1}\} \mathbb{I}\{\mathbb{1}_n^T \theta = 0\}. \quad (3.65)$$

We call (3.65) the spike-and-slab half-Gaussian distribution. Note that the support of the distribution is the intersection of the cone $\{\theta : \theta_1 \le \theta_2 \le ... \le \theta_n\}$ and the subspace $\{\theta : \mathbb{1}_n^T \theta = 0\}$. The parameters v_0 and v_1 play similar roles as in (3.3), which model the closedness between θ_i and θ_{i+1} depending on the value of γ_i.

Proposition 3.6.1. *For any $\gamma \in \{0,1\}^{n-1}$ and $v_0, v_1 \in (0, \infty)$, the spike-and-slab half-Gaussian prior (3.65) is well defined on $\{\theta : \theta_1 \le \theta_2 \le ... \le \theta_n\} \cap \{\theta : \mathbb{1}_n^T \theta = 0\}$, and its density function with respect to the Lebesgue measure restricted on the support is given by*

$$\begin{aligned} p(\theta|\gamma, \sigma^2) &= 2^{n-1} \frac{1}{(2\pi\sigma^2)^{(n-1)/2}} \sqrt{n \prod_{i=1}^{n-1} [v_0^{-1}\gamma_i + v_1^{-1}(1-\gamma_i)]} \\ &\quad \times \exp\left(-\sum_{i=1}^{n-1} \frac{(\theta_{i+1}-\theta_i)^2}{2\sigma^2[v_0\gamma_i + v_1(1-\gamma_i)]}\right) \mathbb{I}\{\theta_1 \le \theta_2 \le ... \le \theta_n\} \mathbb{I}\{\mathbb{1}_n^T \theta = 0\}. \end{aligned}$$

Note that the only place that Proposition 3.6.1 deviates from Proposition 3.2.1 is the extra factor 2^{n-1} due to the isotonic constraint $\{\theta : \theta_1 \le \theta_2 \le ... \le \theta_n\}$ and the symmetry of the density. We complete the model specification by put priors on $\alpha, \gamma, \eta, \sigma^2$ that are given by (3.2), (3.4), (3.5) and (3.6).

Now we are ready to derive the EM algorithm. Since the base graph is a tree, the EM algorithm for reduced isotonic regression is exact. The E-step is given by

$$q_i^{\text{new}} = \frac{\eta \phi(\theta_i - \theta_{i-1}; 0, \sigma^2 v_0)}{\eta \phi(\theta_i - \theta_{i-1}; 0, \sigma^2 v_0) + (1-\eta)\phi(\theta_i - \theta_{i-1}; 0, \sigma^2 v_1)}.$$

The M-step is given by

$$(\alpha^{\text{new}}, \theta^{\text{new}}) = \underset{\alpha, \theta_1 \le \theta_2 \le \ldots \le \theta_n, \mathbb{1}_n^T \theta = 0}{\operatorname{argmin}} F(\alpha, \theta; q^{\text{new}}), \tag{3.66}$$

where

$$F(\alpha, \theta; q) = \|y - \alpha \mathbb{1}_n - \theta\|^2 + \nu\alpha^2 + \sum_{i=1}^{n-1} \left(\frac{q_i}{v_0} + \frac{1 - q_i}{v_1} \right) (\theta_i - \theta_{i-1})^2,$$

and the updates of σ^2 and η are given by (3.17) with $p = n$. The M-step (3.66) can be solved by a very efficient optimization technique. Since $\|y - \alpha\mathbb{1}_n - \theta\|^2 = \|(\bar{y} - \alpha)\mathbb{1}_n\|^2 + \|y - \bar{y}\mathbb{1}_n - \theta\|^2$ by $\mathbb{1}_n^T \theta = 0$, α and θ can be updated independently. It is easy to see that $\alpha^{\text{new}} = \frac{n}{n+\nu}\bar{y}$. The update of θ can be solved by SPAVA [25].

Similar to the Gaussian case, the parameter v_0 determines the complexity of the model. For each v_0 between 0 and v_1, we apply the EM algorithm above to calculate $\widehat{q}$, and then let $\widehat{\gamma}_i = \widehat{\gamma}_i(v_0) = \mathbb{I}\{\widehat{q}_i \ge 1/2\}$ form a solution path. The best model will be selected from the EM-solution path by the limiting version of the posterior distribution as $v_0 \to 0$.

Given a $\gamma \in \{0,1\}^{n-1}$, we write $s = 1 + \sum_{i=1}^{n-1}(1 - \gamma_i)$ to be the number of pieces, and $Z_\gamma \in \{0,1\}^{n \times s}$ is the membership matrix defined in Section 3.3.3. As $v_0 \to 0$, a slight variation of Proposition 3.3.1 implies that θ that follows (3.65) weakly converges to $Z_\gamma \widetilde{\theta}$, where $\widetilde{\theta}$ is distributed by

$$p(\widetilde{\theta}|\gamma, \sigma^2) \propto \exp\left(-\sum_{l=1}^{s} \frac{(\widetilde{\theta}_l - \widetilde{\theta}_{l+1})^2}{2\sigma^2 v_1} \right) \mathbb{I}\{\widetilde{\theta}_1 \le \widetilde{\theta}_2 \le \ldots \le \widetilde{\theta}_s\} \mathbb{I}\{\mathbb{1}_n^T Z_\gamma \widetilde{\theta} = 0\}. \tag{3.67}$$

The following proposition determines the normalizing constant of the above distribution.

Proposition 3.6.2. *The density function of (3.67) is given by*

$$\begin{aligned} p(\widetilde{\theta}|\gamma, \sigma^2) = 2^{s-1}(2\pi\sigma^2)^{-(s-1)/2} \sqrt{\det_{Z_\gamma^T \mathbb{1}_n}(Z_\gamma^T \widetilde{L}_\gamma Z_\gamma)} \times \\ \exp\left(-\sum_{l=1}^{s} \frac{(\widetilde{\theta}_l - \widetilde{\theta}_{l+1})^2}{2\sigma^2 v_1} \right) \mathbb{I}\{\widetilde{\theta}_1 \le \widetilde{\theta}_2 \le \ldots \le \widetilde{\theta}_s\} \mathbb{I}\{\mathbb{1}_n^T Z_\gamma \widetilde{\theta} = 0\}, \end{aligned} \tag{3.68}$$

where Z_γ and $\widetilde{L}_\gamma$ are defined in Section 3.3.3.

Interestingly, compared with the formula (3.23), (3.68) has an extra 2^{s-1} due to the isotonic constraint $\{\widetilde{\theta}_1 \leq ... \leq \widetilde{\theta}_s\}$.

Following Section 3.3.3, we consider a reduced version of the likelihood $y|\alpha, \widetilde{\theta}, \gamma, \sigma^2 \sim N(\alpha \mathbb{1}_n + Z_\gamma \widetilde{\theta}, \sigma^2 I_n)$. Then, with the prior distributions on $\alpha, \widetilde{\theta}, \gamma, \sigma^2$ specified by (3.2), (3.68), (3.4), (3.5) and (3.6), we obtain the joint posterior distribution $p(\alpha, \widetilde{\theta}, \gamma, \sigma^2|y)$. Ideally, we would like to integrate out $\alpha, \widetilde{\theta}, \sigma^2$ and use $p(\gamma|y)$ for model selection. However, the integration with respect to $\widetilde{\theta}$ is intractable due to the isotonic constraint. Therefore, we propose to maximize out $\alpha, \widetilde{\theta}, \sigma^2$, and then the model selection score for reduced isotonic regression is given by

$$g(\gamma) = \max_{\alpha, \widetilde{\theta}_1 \leq ... \leq \widetilde{\theta}_s, \mathbb{1}_n^T Z_\gamma \widetilde{\theta} = 0, \sigma^2} \log p(\alpha, \widetilde{\theta}, \gamma, \sigma^2|y).$$

For each γ, the optimization involved in the evaluation of $g(\gamma)$ can be done efficiently, which is very similar to the M-step updates.

3.7 Numerical Results

In this section, we test the performance of the methods proposed in the paper and compare the accuracy in terms of sparse signal recovery and graphical structure estimation with existing methods. We name our method BayesMSG (Bayesian Model Selection on Graphs) throughout the section. All simulation studies and real data applications were conduced on a standard laptop (2.6 GHz Intel Core i7 processor and 16GB memory) using R/Julia programming language. Detailed codes for implementation of the algorithm are available online at `https://github.com/youngseok-kim/BayesMSG-paper` for reproduction of the results.

Our Bayesian method outputs a subgraph defined by

$$\widehat{\gamma} = \text{argmax} \left\{ g(\gamma) : \gamma \in \{\widehat{\gamma}(v_0)\}_{0 < v_0 \leq v_1} \right\},$$

which is a sub-model selected by the model selection score $g(\gamma)$ on the EM solution path (see Section 3.3.3 for details). Suppose γ^* is the underlying true subgraph that generates the data, we measure the performance of $\widehat{\gamma}$ by false discovery proportion and power. The definitions are

$$\mathsf{FDP} = \frac{\sum_{(i,j)\in E}(1-\widehat{\gamma}_{ij})\gamma^*_{ij}}{\sum_{(i,j)\in E}(1-\widehat{\gamma}_{ij})} \quad \text{and} \quad \mathsf{POW} = 1 - \frac{\sum_{(i,j)\in E}(1-\gamma^*_{ij})\widehat{\gamma}_{ij}}{\sum_{(i,j)\in E}(1-\gamma^*_{ij})},$$

where we adopt the convention that $0/0 = 1$. Note that the above FDP and POW are not suitable for the clustering/biclustering model, because clustering structures are equivalent up to arbitrary clustering label permutations.

The sub-model indexed by $\widehat{\gamma}$ also induces a point estimator for the model parameters. This can be done by calculating the posterior mean of the reduced model specified by the likelihood (3.25) and priors (3.2) and (3.23). With notations $p(y|\alpha, \widetilde{\theta}, \gamma, \sigma^2)$, $p(\alpha|\sigma^2)$ and $p(\widetilde{\theta}|\gamma, \sigma^2)$ for (3.25), (3.2) and (3.23), the point estimator is defined by $\widehat{\beta} = \alpha^{\text{est}} w + Z_{\widehat{\gamma}}\widetilde{\theta}^{\text{est}}$, where Z_γ is the membership matrix defined in Section 3.3.3, and the definition of $(\alpha^{\text{est}}, \widetilde{\theta}^{\text{est}})$ is given by

$$(\alpha^{\text{est}}, \widetilde{\theta}^{\text{est}}) = \underset{\alpha, \widetilde{\theta}\in\{\widetilde{\theta}: w^T Z_{\widehat{\gamma}}\widetilde{\theta}=0\}}{\operatorname{argmax}} \log\left[p(y|\alpha, \widetilde{\theta}, \widehat{\gamma}, \sigma^2)p(\alpha|\sigma^2)p(\widetilde{\theta}|\widehat{\gamma}, \sigma^2)\right],$$

which is a simple quadratic programming whose solution does not depend on σ^2. Note that the definition implies that $\widehat{\beta}$ is the posterior mode of the reduced model. Since the posterior distribution is Gaussian, $\widehat{\beta}$ is also the posterior mean. The performance of $\widehat{\beta}$ will be measured by the mean squared error

$$\mathsf{MSE} = \frac{1}{n}\|X(\widehat{\beta} - \beta^*)\|^2,$$

where β^* is the true parameter that generates the data.

The hyper-parameters a, b, A, B in (3.5) and (3.6) are all set as the default value 1. The same rule is also applied to the extensions in Sections 3.4-3.6.

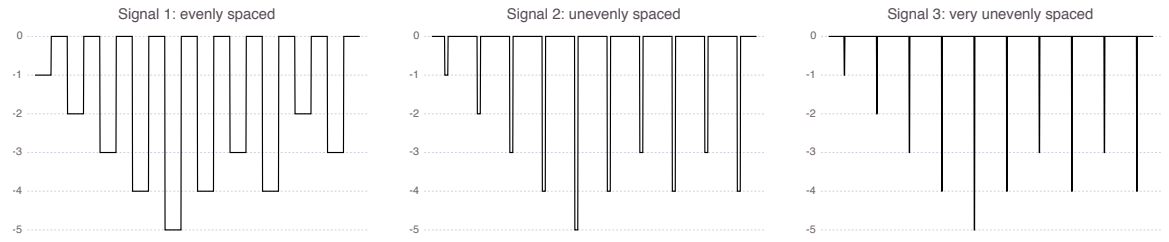


Figure 3.3: Three different signals for the linear chain graph. All signals have 20 pieces. Signal 1 has evenly spaced changes (each piece has length 50), Signal 2 has unevenly spaced changes (a smaller piece has length 10), and Signal 3 has very unevenly spaced changes (a smaller one has length 2).

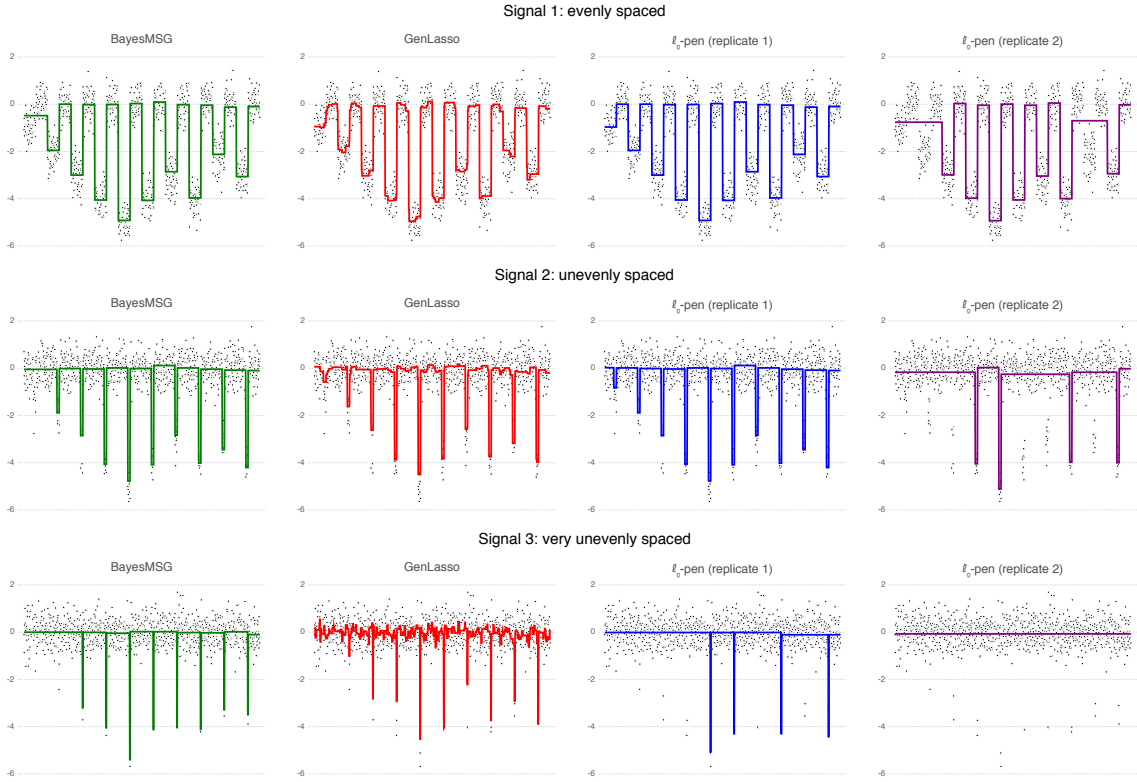


Figure 3.4: Visualization of typical solutions of the three methods when $\sigma = 0.5$. Since ℓ_0-pen is very unstable, we plot contrasting solutions from two independent replicates. *(Top)* Evenly spaced signal; *(Center)* Unequally spaced signal; *(Bottom)* Very unevenly spaced signal; *(Far Left)* BayesMSG; *(Left)* GenLasso; *(Right and Far Right)* Two independent replicates of ℓ_0-pen.

3.7.1 Simulation Studies

In this section, we compare the proposed Bayesian model selection procedure with existing methods in the literature. There are two popular generic methods for graph-structured model selection in the

literature. The first method is the generalized Lasso (or simlply GenLasso henceforth) [152, 136, 154], defined by

$$\widehat{\beta} = \frac{1}{2}\|y - X\beta\|^2 + \lambda \sum_{(i,j)\in E} |\beta_i - \beta_j|. \tag{3.69}$$

The second method is the ℓ_0-penalized least-squares [8, 49, 45], defined by

$$\widehat{\beta} = \frac{1}{2}\|y - X\beta\|^2 + \lambda \sum_{(i,j)\in E} \mathbb{I}\{\beta_i \neq \beta_j\}. \tag{3.70}$$

For both methods, an estimated subgraph is given by

$$\widehat{\gamma}_{ij} = \mathbb{I}\{|\widehat{\beta}_i - \widehat{\beta}_j| \leq \epsilon\},$$

for all $(i, j) \in E$. Here, the number ϵ is taken as 10^{-8}. The two methods are referred to by GenLasso and ℓ_0-pen from now on. In addition to GenLasso and ℓ_0-pen, various other methods [120, 19, 63, 149, 52, 30, 104, 51, 153] that are specific to different models will also be compared in our simulation studies.

Linear Chain Graph

We first consider the simplest linear chain graph, which corresponds to the change-point mdoel explained in Example 3.2.2. We generate data according to $y \sim N(\beta^*, \sigma^2 I_n)$ with $n = 1000$ and $\sigma \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. The mean vector $\beta^* \in \mathbb{R}^n$ is specified in three different cases as shown in Figure 3.3.

We compare the performances of the proposed Bayesian method, GenLasso and ℓ_0-pen. For the linear chain graph, GenLasso is the same as fused Lasso [152]. Its tuning parameter λ in (3.69) is selected by cross validation using the default method of the R package `genlasso` [4]. For ℓ_0-pen, the λ in (3.70) is selected using the method suggested by [45].

The results are summarized in Table 3.1. Some typical solutions of the three methods are plotted in Figure 3.4. In terms of MSE, our Bayesian method achieves the smallest error among the three

Table 3.1: Comparisons of the three methods for the linear chain graph.

	σ	Even			Uneven			Very uneven		
		MSE	FDP	POW	MSE	FDP	POW	MSE	FDP	POW
BayesMSG	0.1	0.00019	0.00	1.00	0.00949	0.00	1.00	0.00217	0.00	0.80
	0.2	0.00585	0.00	0.98	0.01010	0.00	0.97	0.00279	0.00	0.81
	0.3	0.01620	0.01	0.96	0.01116	0.01	0.97	0.00349	0.00	0.81
	0.4	0.01940	0.05	0.95	0.01693	0.02	0.96	0.00837	0.00	0.79
	0.5	0.04667	0.10	0.95	0.03682	0.02	0.96	0.01803	0.05	0.78
GenLasso	0.1	0.00094	0.81	1.00	0.00116	0.90	1.00	0.00570	0.96	1.00
	0.2	0.00374	0.81	1.00	0.00458	0.90	1.00	0.01152	0.94	1.00
	0.3	0.00842	0.81	0.98	0.01024	0.89	1.00	0.02084	0.93	0.99
	0.4	0.01494	0.81	0.98	0.01813	0.88	0.98	0.03376	0.92	0.98
	0.5	0.02345	0.82	0.98	0.02818	0.89	0.98	0.04984	0.92	0.97
ℓ_0-pen	0.1	0.00505	0.00	0.98	0.00288	0.00	0.97	0.02042	0.00	0.81
	0.2	0.00545	0.00	0.98	0.00888	0.00	0.94	0.06049	0.00	0.63
	0.3	0.00399	0.01	0.98	0.00918	0.02	0.94	0.06121	0.00	0.63
	0.4	0.00826	0.02	0.97	0.01119	0.02	0.93	0.06250	0.00	0.63
	0.5	0.06512	0.03	0.92	0.04627	0.02	0.93	0.06452	0.00	0.63

methods when σ is small, and GenLasso has the best performance when σ is large. For model selection performance measured by FDP and POW, the Bayesian method is the best, and ℓ_0-pen is better than GenLasso. We also point out that the solutions of ℓ_0-pen is highly unstable, as shown in Figure 3.4. In terms of computational time, BayesMSG, GenLasso and ℓ_0-pen require 5.2, 11.8 and 19.0 seconds on average.

It is not surprising that GenLasso achieves the lowest MSE in the low signal strength regime. This is because Lasso is known to produce estimators with strong bias towards zero, and therefore it is favored when the true parameters are close to zero. The other two methods, BayesMSG and ℓ_0 are designed to achieve nearly unbiased estimators when the signals are strong, and therefore show their advantages over GenLasso when the signal strength is large.

An interesting question would be if it is possible to design a method that works well in all of the three criteria (MSE, POW, FDP) with both low and strong signal? Unfortunately, a recent paper [139] proves that this is impossible. The result of [139] rigorously establishes the incompatibility phenomenon between selection consistency and rate-optimality in high-dimensional sparse regression. We therefore believe our proposed BayesMSG, which performs very well in terms of the three

criteria (MSE, POW, FDP) except for the MSE in the low signal regime is a very good solution in view of this recent impossibility result.

Regression with Graph-structured Coefficients

For this experiment, we consider a linear regression setting with graph structured sparsity on the regression coefficients. We sample random Gaussian measurements $X_{ij} \sim N(0,1)$ and measurement errors $\epsilon_i \sim N(0,1)$. Then we construct a design matrix $X \in \mathbb{R}^{n \times p}$ and a response vector $y = X\theta + \epsilon \in \mathbb{R}^n$, where θ is a vector of node attributes of an underlying graph G.

We fix $n = 500$ throughout this simulation study, and consider three different graph settings listed in Table 3.2. When G is a star graph with the center node at 0, our proposed model in Section 3.2 corresponds to the sparse linear regression problem. We compare our proposed approach BayesMSG with the following baseline methods implemented in R programming language: Lasso (`glmnet` R package) [48], and Bayesian Spike-And-Slab linear regression (BSAS) via MCMC (`BGLR` R package) [120]. All the R packages listed here are implemented using their default setting and their recommended model selection methods.

Next, when G is a linear chain graph, the model corresponds to the linear regression problem with a sparse graph difference vector $(\theta_2 - \theta_1, \cdots, \theta_p - \theta_{p-1})$. This problem setting is particularly studied for fused Lasso [152] and the approximate ℓ_0 regression setting (ITALE) [173]. Therefore, we compare BayesMSG with the above baseline approaches: fused Lasso (`genlasso` R package) and ITALE (`ITALE` R package) on the linear chain graph.

Finally, when G is a complete graph, the model corresponds to the linear regression problem with clustered coefficients, i.e. β_j's may be clustered together. This particular problem setting is also considered in the studies of GenLasso [155] and OSCAR [19]. We compare BayesMSG with the following baseline methods: GenLasso and OSCAR (`lqa` R package). In brief, OSCAR seeks to solve

$$\text{minimize}_\beta \quad \frac{1}{2}\|y - X\theta\|_2^2 + \lambda \sum_{j=1}^{p} (c(j-1)+1)|\theta|_{(j)}.$$

Table 3.2: Simulation Settings for Gaussian Design.

Graph	# of nodes p	# of edges m	# of fixed nodes	Description
Star	1,001	1,001	1	regression with sparse coefficients
Linear chain	1,000	999	0	regression with gradient-sparse signals
Complete	200	19,900	0	regression with clustered coefficients

Table 3.3: Simulation Results for Gaussian Design $C = 0.5$ (above) $C = 1$ (below).

Graph	Star Graph			Linear Chain Graph			Complete Graph		
Method	BMSG	Lasso	BSAS	BMSG	GenLasso	ITALE	BMSG	GenLasso	OSCAR
MSE	0.579	0.792	0.530	0.099	0.276	0.159	0.399	0.472	0.438
Time	5.425	1.663	11.32	3.161	5.791	6.543	27.52	50.62	18.18
FDP	0.324	0.662	-	0.000	0.953	0.106	0.224	0.614	0.467
POW	0.978	0.995	-	0.954	0.980	0.988	0.996	1.000	1.000

Graph	Star Graph			Linear Chain Graph			Complete Graph		
Method	BMSG	Lasso	BSAS	BMSG	GenLasso	ITALE	BMSG	GenLasso	OSCAR
MSE	0.624	0.789	0.592	0.099	0.274	0.176	0.403	0.472	0.442
Time	5.396	1.721	12.75	3.436	5.904	6.554	22.85	51.42	15.80
FDP	0.319	0.658	-	0.000	0.930	0.032	0.205	0.290	0.208
POW	0.983	0.982	-	0.994	0.988	0.996	0.998	1.000	0.994

A true graph-structured sparse signal is constructed as follows. For the case of star graphs, $\theta_j^* = 0.5C$ for $j = 1, \cdots, 40$ and 0 otherwise. For the cases of linear change graphs and complete graphs, $\theta_j^* = C$ for $j = 1, \cdots, 0.4p$, $\theta_j^* = 2C$ for $j = 0.4p + 1, \cdots, 0.7p$, $\theta_j^* = 3C$ for $j = 0.7p + 1, \cdots, 0.9p$ and $4C$ otherwise. Tables 3.3 displays the estimation error (MSEs) $\|\theta^* - \widehat{\theta}\|_2$ on the test data sets, and computation times. BMSG, BSAS and GenLasso are abbreviations for BayesMSG, Bayesian Spike-And-Slab regression and Generalized Lasso. Each reported error value is averaged across 10 independent simulations with different random seeds.

The results show that our proposed BayesMSG method is the overall winners across all models in both estimation error (MSE) and model selection error (FDP and POW). The advantage is especially obvious for the linear chain graph and the complete graph. The only case where BayesMSG cannot beat its competitor (BSAS) is the estimation error in the star graph case (sparse linear regression). However, we note that BSAS is an MCMC-based method that does not involve model selection but perform Bayesian model averaging. Thus, the solution of BSAS is not sparse. On the other hand, BayesMSG is designed for model selection, and therefore performs much better in terms of model

selection error (FDP and POW).

Two-Dimensional Grid Graph

We consider the two-dimensional grid graph described in Example 3.2.3. The data is generated according to $y_{ij} \sim N(\kappa\mu^*_{ij}, 1)$ for $i = 1, ..., 21$ and $j = 1, ..., 21$, where

$$\mu^*_{ij} = \left\lceil 2.8 \cos\left(\frac{\sqrt{i^2 + j^2}}{2\pi}\right) - 0.2 \right\rceil,$$

and $\kappa \in \{1, 2, ..., 10\}$ is used to control the signal strength. Note that μ^*_{ij} has a piecewise constant structure because of the operation by $\lceil\cdot\rceil$ that denotes the integer part. In fact, μ^*_{ij} only takes 5 possible values as shown in Figure 3.6.

Since the R package `genlasso` does not provide a tuning method for the λ in (3.69) for the two-dimensional grid graph setting, we report MSE based on the λ selected by cross validation, and FDP and POW are reported based on the λ that minimizes the FDP. The λ in ℓ_0-pen is tuned by the method in [45].

The results are shown in Figure 3.5. It is clear that our method outperforms the other two in terms of all the evaluation criteria when the signal strength is not very small. When the signal strength is very small, GenLasso achieves the lowest MSE but shows poor model selection performance. We also illustrate the solution path of our method in Figure 3.6. Typical solutions of GenLasso and ℓ_0-pen are visualized in Figure 3.7. We observe that ℓ_0-pen tends to oversmooth the data, while GenLasso tends to undersmooth. In terms of the computational time, BayesMSG, GenLasso and ℓ_0-pen require 21.2, 26.7 and 8.4 seconds on average.

Generic Graphs

In this section, we consider some graphical structures that naturally arise in real world applications. The three graphs to be tested are the Chicago metropolitan area road network[2], the Enron email

2. `http://www.cs.utah.edu/~lifeifei/SpatialDataset.htm`

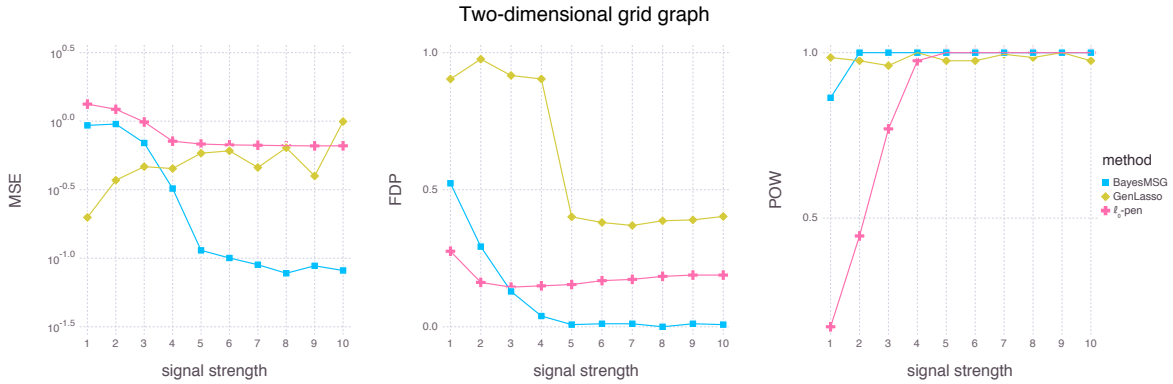


Figure 3.5: Comparison of the three methods for the two-dimensional grid graph. (*Left*) MSE; (*Center*) FDP; (*Right*) POW.

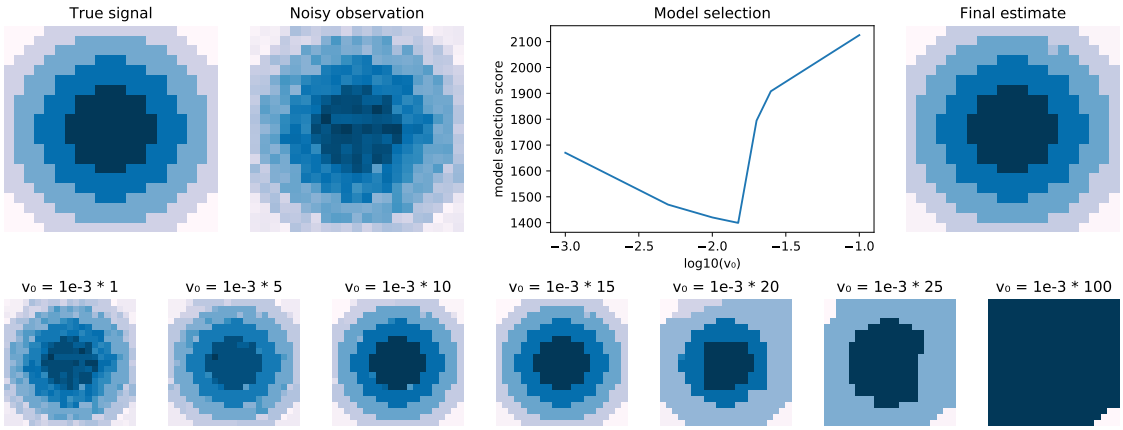


Figure 3.6: (*Top panels*) True signal, noisy observations, model selection score, and final estimate; (*Bottom panels*) A regularization path from $v_0 = 10^{-3}$ to $v_0 = 10^{-1}$.

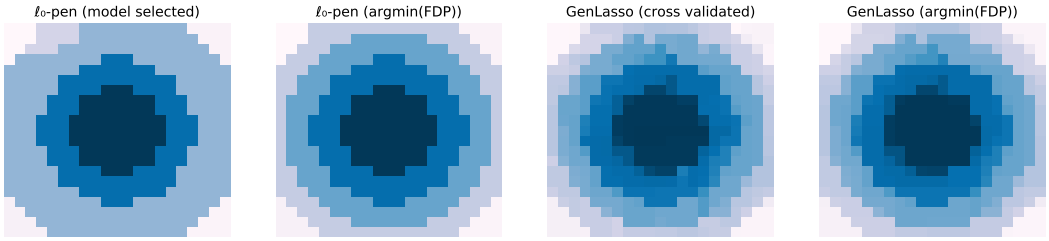


Figure 3.7: *(Far Left)* ℓ_0-pen with λ selected using the method in [45]; *(Left)* ℓ_0-pen with λ that minimizes FDP; *(Right)* GenLasso with λ selected by cross validation; *(Far Right)* GenLasso with λ that minimizes FDP.

Table 3.4: Graph properties of the three real networks.

Name	# of nodes	# of edges	mean.ER	sd.ER	diameter	# of CC
Chicago roadmap	4126	4308	0.9575	0.0499	324	1
Enron email	4112	14520	0.2831	0.2341	14	1
Facebook egonet	4039	88234	0.0457	0.0608	8	1

Table 3.5: Important features of the signals on the three networks.

Name	# of clust	# of nodes in each clusters	# of cuts	total variation
Chicago roadmap	4	$(576, 678, 835, 2037)$	31	$31 \times \kappa$
Enron email	4	$(384, 538, 1531, 1659)$	4570	$5047 \times \kappa$
Facebook egonet	4	$(750, 753, 778, 1758)$	651	$1220 \times \kappa$

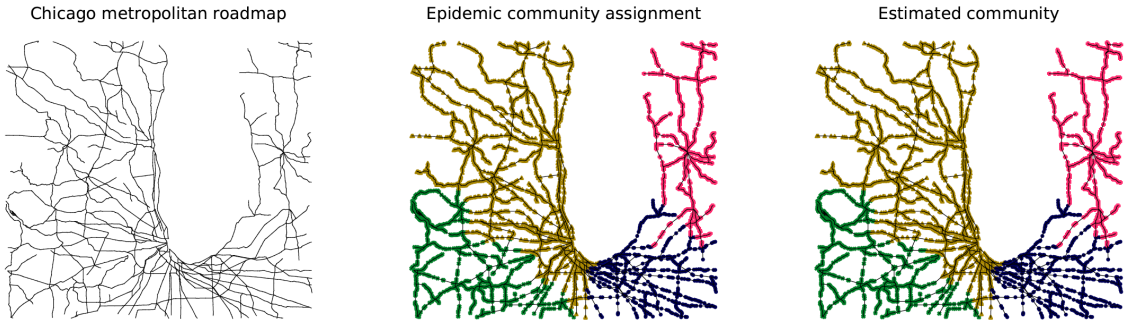


Figure 3.8: The Chicago roadmap network with signals exhibiting four clusters.

network[3], and the Facebook egonet network[4]. For all the three networks, we extract induced subgraphs of sizes about 4000. Graph properties for the three networks are summarized in Table 3.4. For each network, we calculate its number of nodes, number of edges, mean and standard deviation of effective resistances, diameter, and number of connected components. We observe that the three networks behave very differently. The Chicago roadmap network is locally and globally tree-like, since its number of edges is very close to its number of nodes, and the distribution of its effective resistances highly concentrates around 1. The other two networks, the Enron email network and the Facebook egonet, are denser graphs but their effective resistances behave in very different ways.

For each network, we generate data according to $y_i \sim N(\kappa\mu_i^*, 1)$ on its set of nodes, with the signal strength varies according to $\kappa \in \{1, 2, ..., 5\}$. The signal μ^* for each graph is generated as

3. `http://snap.stanford.edu/data/email-Enron.html`

4. `http://snap.stanford.edu/data/ego-Facebook.html`

follows:

1. Pick four anchor nodes from the the set of all nodes uniformly at random.

2. For each node, compute the the length of the shortest path to each of the four anchor nodes.

3. Code the ith node by j if the jth anchor node is the closest one to the ith node. This gives four clusters for each graph.

4. Generate a piecewise constant signal $\mu_i^* = j$.

Some properties of the signals are summarized in Table 3.5, where the number of cuts of μ^* with respect to the base graph $G = (V, E)$ is defined by $\sum_{(i,j)\in E} \mathbb{I}\{\mu_i^* \neq \mu_j^*\}$, and the total variation of μ^* means $\sum_{(i,j)\in E} |\mu_i^* - \mu_j^*|$. We also plot the signal on the Chicago roadmap network in Figure 3.8.

Since the R package `genlasso` does not provide a tuning method for the λ in (3.69) for a generic graph, we report MSE based on the λ selected by cross validation, and FDP and POW are reported based on the λ that minimizes the FDP. The λ in ℓ_0-pen is tuned by the method in [45].

The results are shown in Figure 3.9. It is clear that our method outperforms the other two. When the signal strength κ is small, we observe that GenLasso sometimes has the smallest MSE, but its MSE grows very quickly as κ increases. For most κ's, our method and ℓ_0-pen are similar in terms of MSE. In terms of the model selection performance, GenLasso is not competitive, and our method outperforms ℓ_0-pen.

Comparison of Different Base Graphs

One key ingredient of our Bayesian model selection framework is the specification of the base graph. For the same problem, there can be multiple ways to specify the base graph that lead to completely different models and methods. In this section, we consider an example and compare the performances of different Bayesian methods with different base graphs.

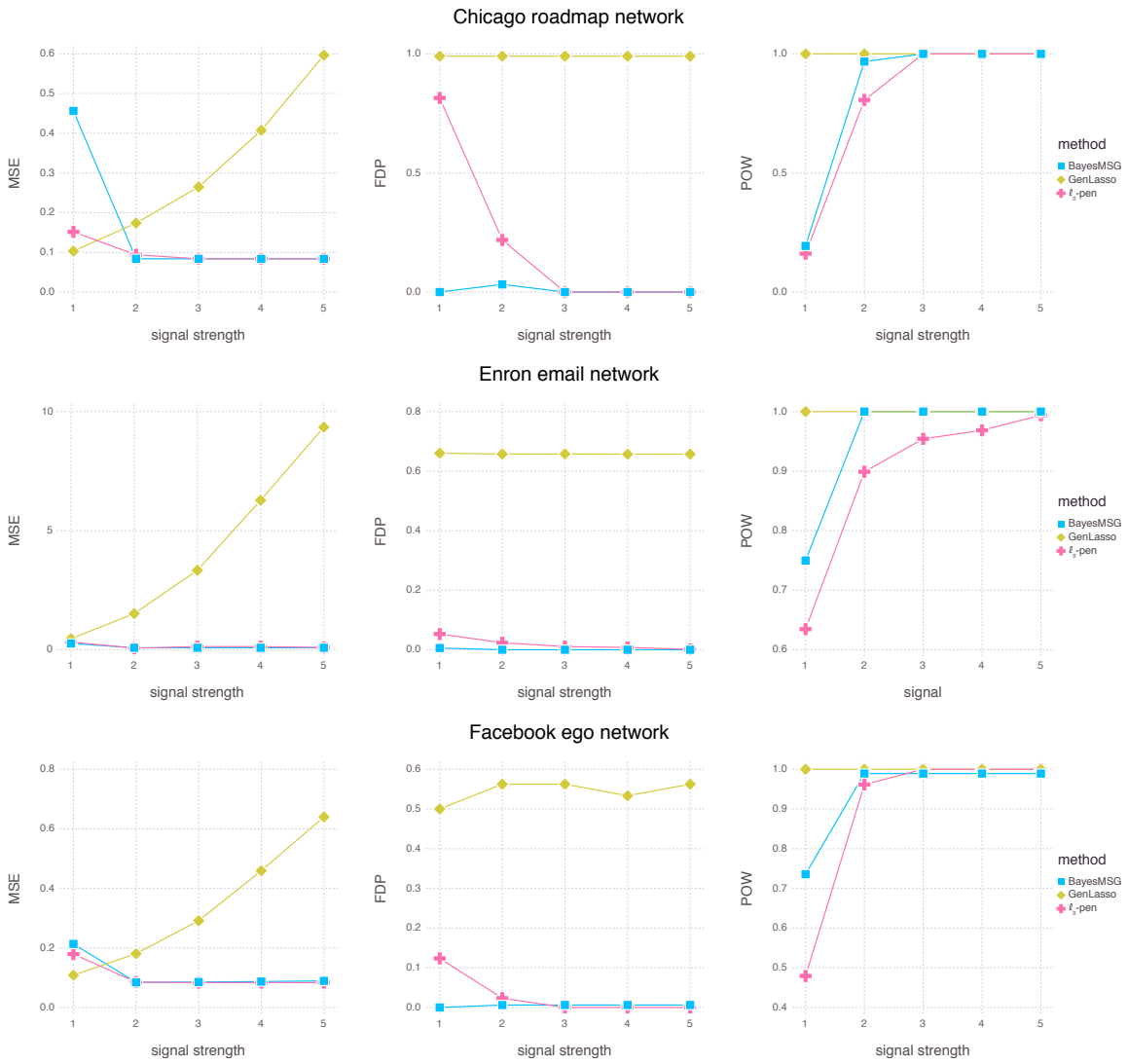


Figure 3.9: Comparison of the three methods on generic graphs. (*Top*) Chicago Roadmap network; (*Center*) Enron Email network; (*Bottom*) Facebook Ego network.

We consider observations $y_{ij} \sim N(\theta^*_{ij}, 1)$ for $i \in [n_1]$ and $j \in [n_2]$. We fix $n_2 = 12$ and vary n_1 from 24 to 144. The signal matrix $\theta^* \in \mathbb{R}^{n_1 \times n_2}$ has a checkerboard structure as shown in Figure 3.10. That is, the $n_1 \times n_2$ matrix is divided into 6×6 equal-sized blocks. On the (u, v)th block, $\theta^*_{ij} = 2(u + v - 6)$.

The following models are considered to fit the observations:

1. *Vector clustering.* We regard the matrix $y \in \mathbb{R}^{n_1 \times n_2}$ as a $n_1 n_2$-dimensional vector and apply the clustering model described in Section 3.4.2 with $n = k = n_1 n_2$.

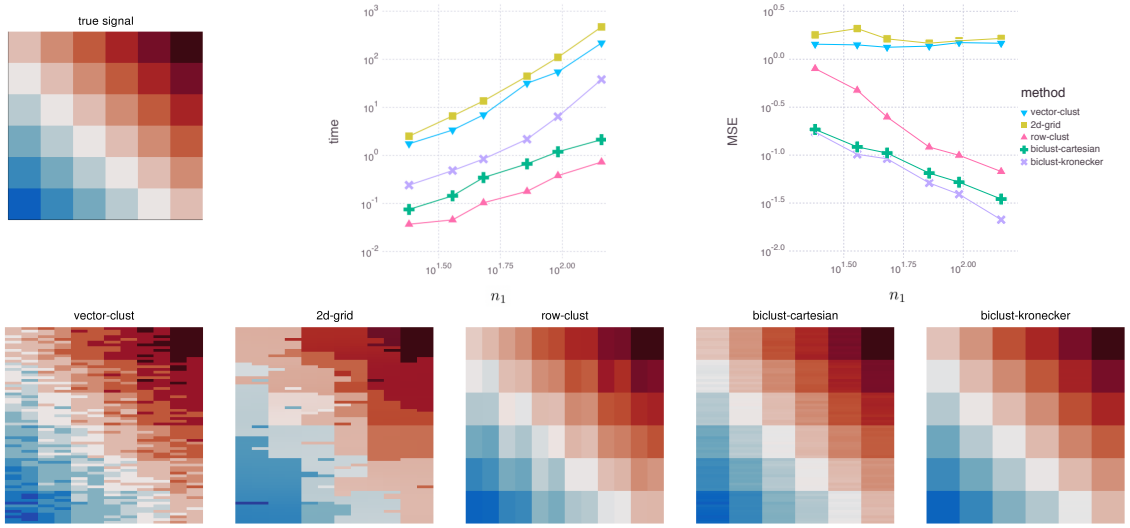


Figure 3.10: (*Top left*) True signal; (*Top center*) Computational time; (*Top right*) MSE; *(Bottom)* Heatmaps of estimators using different models ($n_1 = 72$).

2. *Two-dimensional grid graph.* The two-dimensional image denoising model described in Example 3.2.3 is fit to the observations.

3. *Row clustering.* We regard the matrix $y \in \mathbb{R}^{n_1 \times n_2}$ as $n = n_1$ observations in $\mathbb{R}^d$ with $d = n_2$, and then fit the clustering model described in Section 3.4.2 to the rows of y with $n = k = n_1$.

4. *Cartesian product biclustering.* The biclustering model induced by the Cartesian product described in Section 3.5.3 is fit to the observations.

5. *Kronecker product biclustering.* The biclustering model induced by the Kronecker product described in Section 3.5.3 is fit to the observations.

Figure 3.10 summarizes the results. In terms of MSE, the vector clustering and the two-dimensional grid graph do not fully capture the structure of the data and thus perform worse than all other methods. Both the biclustering models are designed for the checkerboard structure, and they therefore have the best performances. Between the two biclustering models, the one induced by the Kronecker product has a smaller MSE at the cost of a higher computational time.

To summarize the comparisons, we would like to emphasize that the right choice of the base

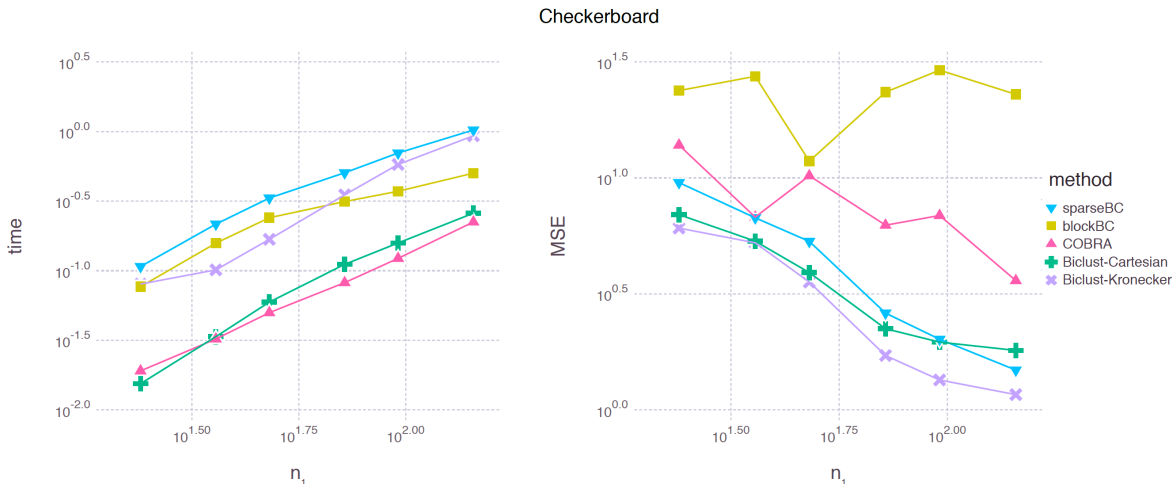


Figure 3.11: Comparisons of the 5 biclustering methods for the checkerboard data. (*Left*) Time; (*Right*) MSE.

graph has an enormous impact to the result. This also highlights the flexibility of our Bayesian model selection framework that is able to capture various degrees of structures of the data.

Biclustering

To evaluate performance of our biclustering methods in comparison to existing methods, we provide a simulation study using the same data in Section 3.7.1. The true parameter $\theta \in \mathbb{R}^{n_1 \times n_2}$ has a checkerboard structure and is visualized in Figure 3.10. More precisely, we consider observations $y_{ij} \sim N(\theta^*_{ij}, 1)$ for $i \in [n_1]$ and $j \in [n_2]$. We fix $n_2 = 12$ and vary n_1 from 24 to 144. The signal matrix $\theta^* \in \mathbb{R}^{n_1 \times n_2}$ has a checkerboard structure as shown in Figure 3.10. That is, the $n_1 \times n_2$ matrix is divided into 6×6 equal-sized blocks. On the (u, v)th block, $\theta^*_{ij} = 2(u + v - 6)$.

For comparison, we consider the three competitors, blockBC [63], sparseBC [149], and COBRA [30], with the implementation via R packages `blockcluster`, `sparseBC` and `cvxbiclustr`. In summary, blockBC and sparseBC are based on the minimization of $\sum_{i,j}(y_{ij} - \mu_{z_1(i)z_2(j)})^2$ given the number k_1 and k_2 of row and column clusters, respectively, where $\mu \in \mathbb{R}^{k_1 \times k_2}$ is a matrix of latent hidden bicluster means and z_1 and z_2 are row and column cluster assignments, respectively. The methods blockBC and sparseBC use different approaches for the estimation of row and column clusters, i.e. blockBC uses a block EM algorithm. sparseBC solves a penalized linear regression

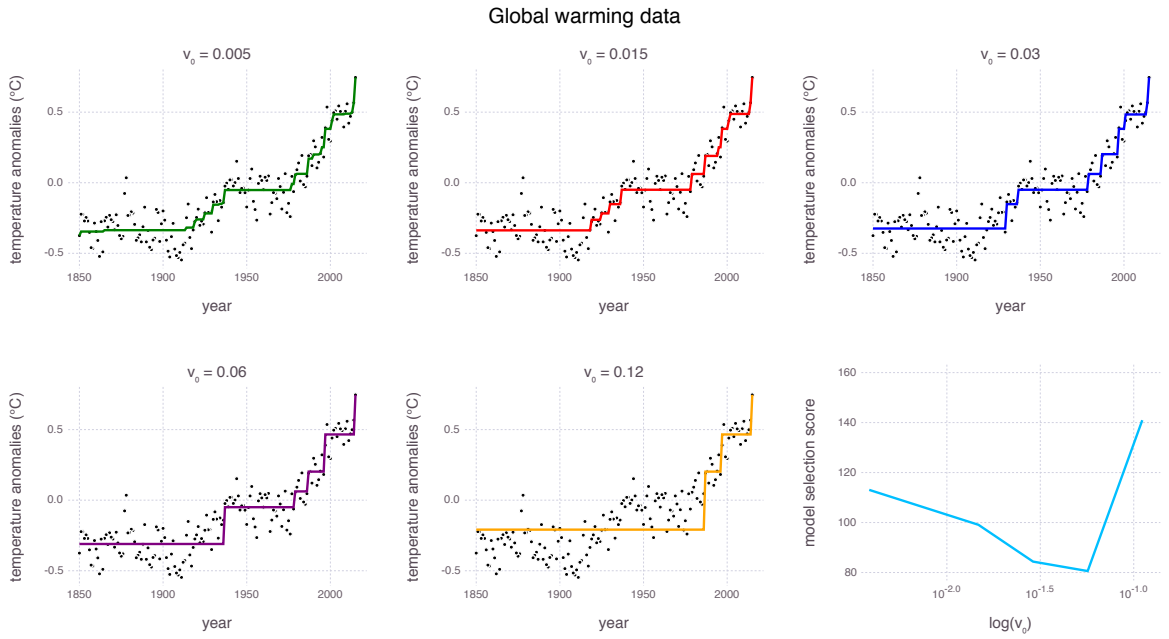


Figure 3.12: The solution path (smaller v_0 at top left and larger v_0 at bottom right) for Bayesian reduced isotonic regression.

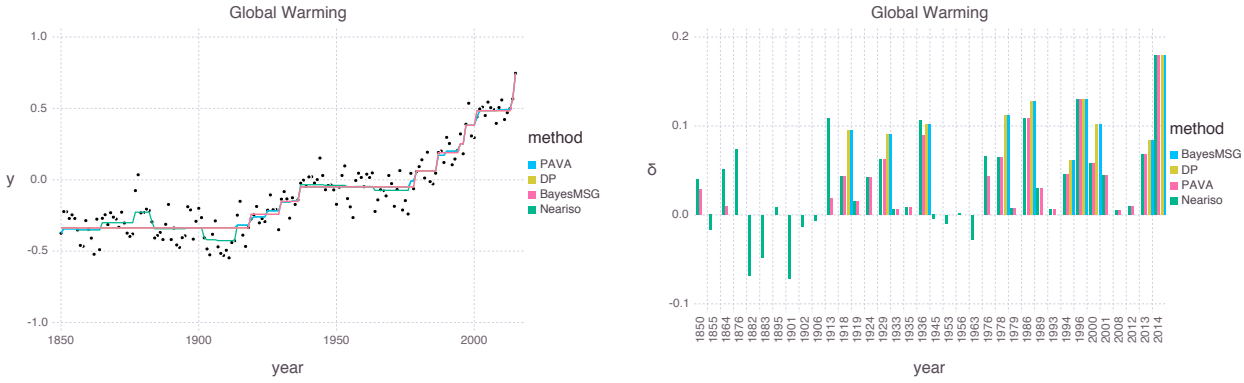


Figure 3.13: Comparison of various isotonic regression methods. (*Left*) The estimated isotonic signals (PAVA, DP, BayesMSG, NearIso); let us mention that DP and BayesMSG signals exactly coincide. (*Right*) The estimated differences between the two adjacent years; the only years (x-axis) with at least one nonzero differences are reported.

with the ℓ_1-penalty $\lambda\|\mu\|_1$. COBRA is an alternating direction method of multipliers (ADMM) algorithm based on the minimization of $\sum_{i,j}(y_{ij}-\theta_{ij}))+\text{pen}_{\text{row}}(\theta)+\text{pen}_{\text{col}}(\theta)$ where $\text{pen}_{\text{row}}(\theta) = \sum_{i<j} w_{ij}\|\theta_{i*} - \theta_{j*}\|_2$ and pen_{col} is defined similarly.

The result is displayed in Figure 3.11. The result shows that BayesMSG Kronecker outperforms other methods in terms of MSE. BayesMSG Cartesian also produces competitive solutions in terms of MSE within short amount of computation time. Note that BayesMSG and COBRA are regularization path based methods, and require less computation time than other approaches (blockBC and sparseBC) based on cross validation.

3.7.2 Real Data Applications

In this section, we apply our methods to three different data sets.

Global warming data

The global warming data has been studied previously by [170, 153]. It consists of 166 data points in degree Celsius from 1850 to 2015. Here we fit the Bayesian reduced isotonic regression discussed in Section 3.6. Our results are shown in Figure 3.12.

When v_0 is nearly zero, the solution is very close to the regular isotonic regression that can be solved efficiently by the pool-adjacent-violators algorithm (PAVA) [104]. When $v_0 = 0.005$, we obtain a fit with 24 pieces. The PAVA outputs a very similar fit also with 24 pieces. In contrast, the Bayesian model selection procedure suggests a model with $v_0 = 0.06$, which has only 6 pieces, a significantly more parsimonious and a more interpretable fit. This may suggest global warming is accelerating faster in recent years. The same conclusion cannot be obtained from the suboptimal fit with 24 pieces.

To compare with existing methods, we have implemented reduced isotonic regression with dynamic programming (DP) algorithm introduced by [51] and the near isotonic (NearIso) regression [153]. Figure 3.13 shows the estimated signals (left panel) and the estimated differences $\{\theta_{i+1} - \theta_i : i = 1, \cdots, n-1\}$ between the two adjacent years (right panel). Since the DP algorithm does not have a practical model selection procedure, we use the number of pieces $k = 9$ selected by BayesMSG. NearIso uses Mallow's C_p for model selection [153]. The left panel of Figure 3.13 shows that BayesMSG outputs a more sparse but still reasonable isotonic fit. On the other hand,

NearIso relaxes the isotonic constraint, allowing non-monotone signals but penalizing the decreasing portion of variations. Indeed, one can see from the right panel of Figure 3.13 that the changepoints of NearIso contain those of PAVA. The PAVA solution is the least parsimonious isotonic fit by definition, and thus we can conclude that NearIso does not seem to find a more parsimonious fit.

Lung cancer data

We illustrate the Bayesian biclustering models by a gene expression data set from a lung cancer study. The same data set has also been used by [14, 94, 137, 30]. Following [30], we study a subset with 56 samples and 100 genes. The 56 samples comprise 20 pulmonary carcinoid samples (Carcinoid), 13 colon cancer metastasis samples (Colon), 17 normal lung samples (Normal) and 6 small cell lung carcinoma samples (SmallCell). We also apply the row and column normalizations as has been done in [30].

Our goal is to identify sets of biologically relevant genes, for example, that are significantly expressed for certain cancer types. We fit both Bayesian biclustering models (Section 3.5.3) induced by the Cartesian and Kronecker products to the data with $n_1 = 56$, $n_2 = 100$, $k_2 = 10$, and $k_2 = 20$. Recall that k_1 and k_2 are upper bounds of the numbers of row and column clusters, and the actual numbers of row and column clusters will be learned through Bayesian model selection. To pursue a more flexible procedure of model selection, we use two independent pairs of (v_0, v_1) for the row structure and the column structure. To be specific, let (v_0, v_1) be the parameters for the row structure, and the parameters for the column structure are set as (cv_0, cv_1) with some $c \in \{1/10, 1/5, 1/2, 1, 2, 5, 10\}$. Then, the model selection scores are computed with both v_0 and c varying in their ranges.

The results are shown in Figure 3.14. The two methods select different models with different interpretations. The Cartesian product fit gives 4 row clusters and 8 column clusters, while the Kroneker product fit gives 6 row clusters and 11 column clusters. Even though we have not used the information of the row labels for both biclustering methods, the row clustering structure output by the Cartesian product model almost coincides with these labels except one. On the other hand,

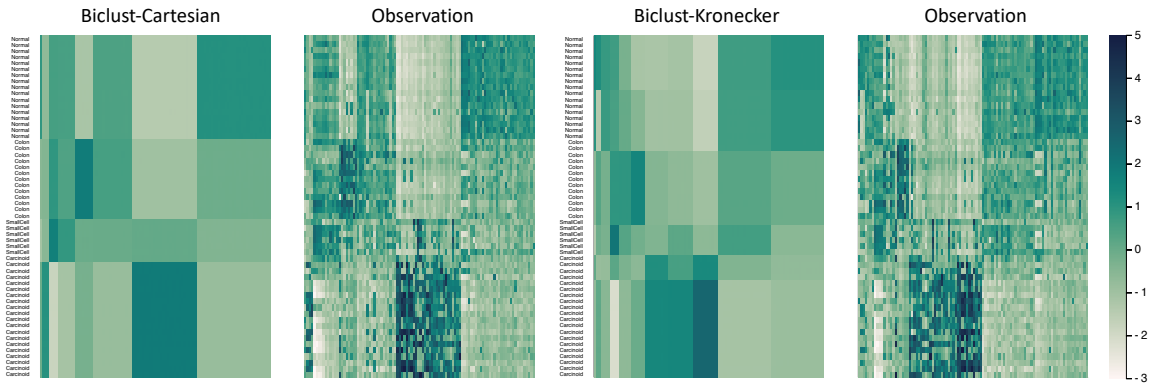


Figure 3.14: Results of biclustering for the lung cancer data[5].

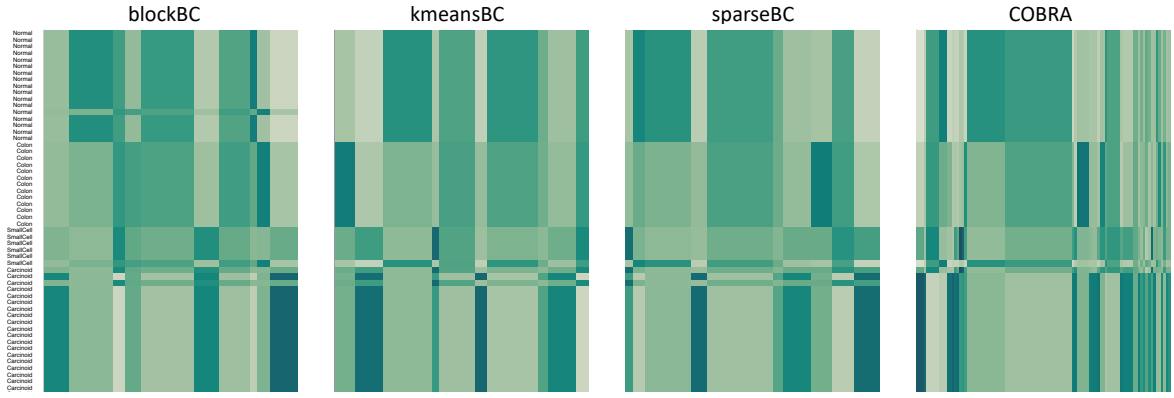


Figure 3.15: Comparison of existing biclustering methods.

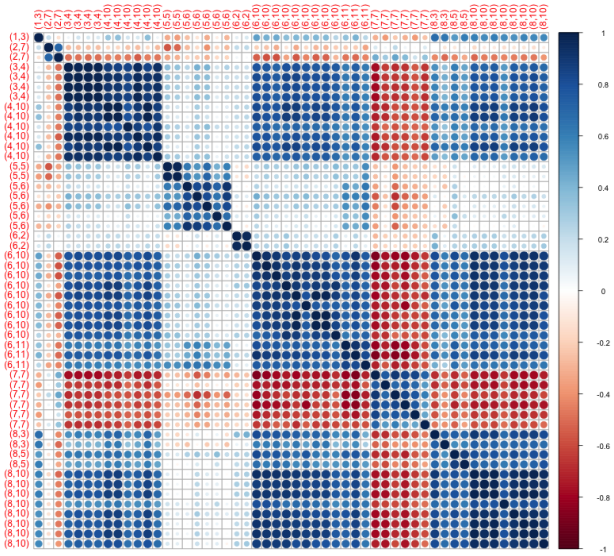


Figure 3.16: A correlation plot of the selected genes.

Table 3.6: Description or GenBank ID of the selected gene clusters of size at least 2 and at most 5

Cluster labels	Gene description/GenBank ID
$(2, 7)$	"proteoglycan 1, secretory granule", "AI932613: Homo sapiens cDNA, 3 end"
$(3, 4)$	"AI147237: Homo sapiens cDNA, 3 end", "S71043: Ig A heavy chain allotype 2", "advanced glycosylation end product-specific receptor", "leukocyte immunoglobulin-like receptor, subfamily B"
$(5, 5)$	'immunoglobulin lambda locus", "glypican 3"
$(5, 6)$	"glutamate receptor, ionotropic, AMPA 2", "small inducible cytokine subfamily A", "W60864: Homo sapiens cDNA, 3 end", "secreted phosphoprotein 1", "LPS-induced TNF-alpha factor"
$(6, 2)$	"interleukin 6", "carcinoembryonic antigen-related cell adhesion molecule 5"
$(6, 11)$	"secretory granule, neuroendocrine protein 1", "alcohol dehydrogenase 2", "neurofilament, light polypeptide"
$(8, 3)$	"fmajor histocompatibility complex, class II", "glycoprotein (transmembrane) nmb"
$(8, 5)$	"N90866: Homo sapiens cDNA, 3 end", 'receptor (calcitonin) activity modifying protein 1"

the Kronecker product model leads to a finer row clustering structure, with potential discoveries of subtypes of both normal lung samples and pulmonary carcinoid samples.

Using the same lung cancer data set, we have also compared the BayesMSG Cartesian product method with several existing biclustering methods implemented via R packages `blockcluster`, `sparseBC` and `cvxbiclustr`. Figure 3.11 displays the final estimates of the four competitors, blockBC [63], sparseBC [149], kmeansBC [52] and COBRA [30]. One can qualitatively compare these results with the BayesMSG solutions in Figure 3.15.

The results show that BayesMSG and all the competitors select models with $\widehat{k}_1 = 4$ row clusters. However, BayesMSG selects a smallest number of column clusters. This implies that BayesMSG favors a more parsimonious model compared to the others. Also, the selected BayesMSG Cartesian model achieves a lowest misclassification error for the row (cancer type).

An important goal of biclustering is to simultaneously identify gene and tumor types. To be specific, we seek to find genes that show different expression levels for different types of samples. To this end, we report those genes that are clustered together by both the Cartesian and Kronecker product structures. Groups of genes with size between 2 and 5 are reported in Table 3.6. Note that our gene clustering is assisted by the sample clustering in the biclustering framework, which is different from gene clustering methods that are only based on the correlation structure [14]. As a sanity check, the correlation matrix of the subset of the selected genes is plotted in Figure 3.16, and

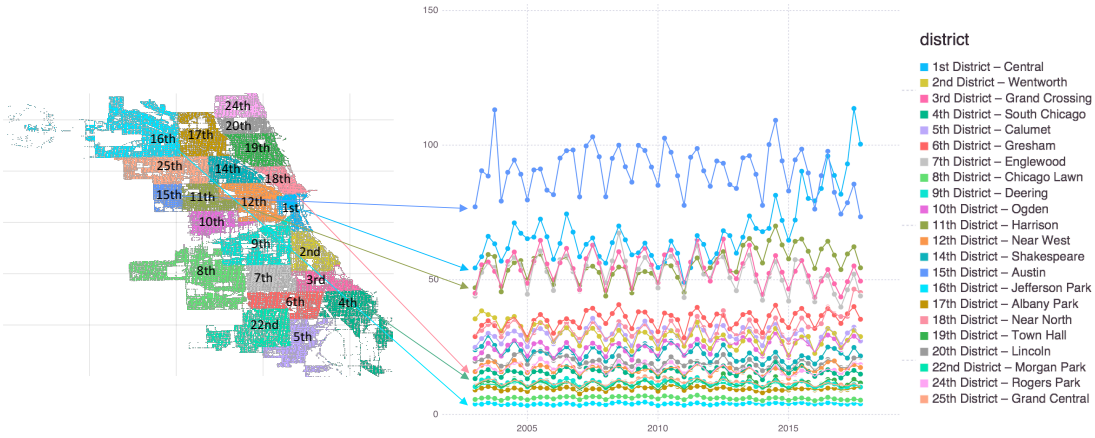


Figure 3.17: Visualization of the Chicago crime data after preprocessing.

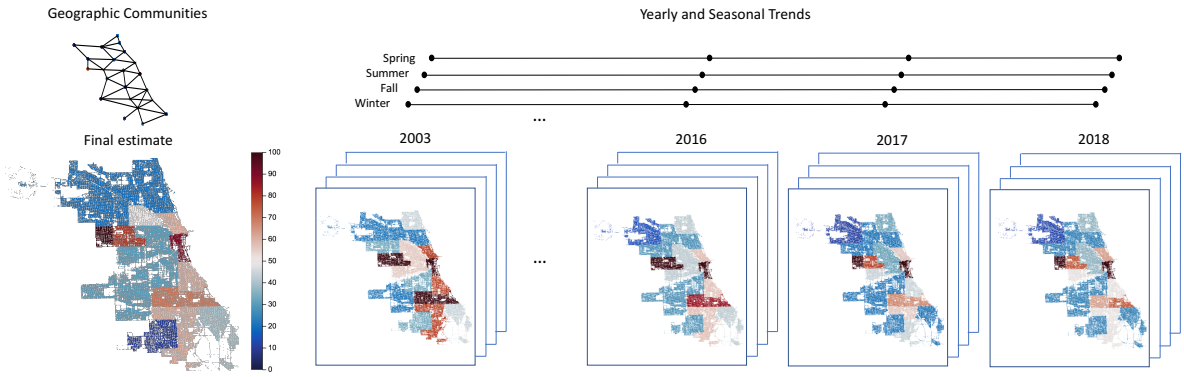


Figure 3.18: Visualization of Bayesian model selection for the Chicago crime data. (*Left*) The overall geographical pattern; (*Right*) Four different patterns from 2003 to 2018.

we can observe a clear pattern of block structure.

Chicago crime data

The Chicago crime data is publicly available at Chicago Police Department website[6]. The report in the website contains time, type, district, community area, latitude and longitude of each crime occurred. After removing missing data, we obtain 6.0 millions of crimes that occurred in 22 police districts from 2003 to 2018 (16 years). Here we restrict ourselves to the analysis of the spatial and

5. The rows of the four heatmaps are ordered in the same way according to the labels of tumor types.

6. `https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/`

temporal structure of Chicago crimes within the past few years, ignoring the types or categories of the crimes.

Since the 22 districts have different area sizes, we divide the total numbers of crimes in each district by its population density (population per unit area). We will call this quantity the *Chicago crime rate*. We observe that the Chicago crime rates exhibit decreasing patterns over the years. Since our study is focused on the relative comparisons among different police districts, we divide each entry by the sum of the yearly Chicago crime rates over all the 22 district in its current year. We will call this quantity the *relative crime rate*. Admittedly this preprocessing step does not reflect the difference between the *residential* and the *floating* populations in each district which might be important for the analysis of the crime data. For instance, around O'Hare international airport, it is very likely that the floating and the residential populations differ a lot. After the preprocessing, we obtain a three-way tensor with size $22 \times 16 \times 4$, for 22 districts, 16 years, and 4 seasons, which is visualized in Figure 3.17.

Our main interest is to understand the geographical structure of the relative crime rates and how the structure changes over the year. A Bayesian model is constructed for this purpose by using the graphical tools under the proposed framework. We define a graph characterizing the geographical effect by $G_1 = (V_1, E_1)$ with the vertex set $V_1 = \{1, 2, \cdots, 22\}$ and the edge set $E_1 = \{(i, j) : \text{the } i\text{-th and } j\text{-th districts are adjacent}\}$. A graph characterizing the temporal effect is given by $G_2 = (V_2, E_2)$ with $V_2 = \{1, 2, ..., 16\}$ and $E_2 = \{(i, i + 1) : i = 1, ..., 15\}$. Then, the $22 \times 16 \times 4$ tensor is modeled by a spike-and-slab Laplacian prior with the base graph $G_1 \square G_2$, in addition to a multivariate extension (Section 3.4.1) along the dimension of 4 seasons.

The result of the Bayesian model selection for the Chicago crime data is visualized in Figure 3.18. The geographical structure of the relative crime rates exhibit four different patterns according to the partition $\{2003, 2004, ..., 2015\}$, $\{2016\}$, $\{2017\}$, $\{2018\}$. While geographical compositions of the crimes are similar from 2003 to 2015, our results reveal that the last three years have witnessed dramatic changes. In particular, in these three years, the relative crime rates of Districts 11 and 15 were continuously decreasing, and the relative crime rates of Districts 1 and 18 show the opposite

trend. This implies that the overall crime pattern is moving away from historically dangerous areas to downtown areas in Chicago.

3.8 Supplementary Material

3.8.1 Some Basics on Linear Algebra

For a symmetric matrix Γ with rank r, it has an eigenvalue decomposition $\Gamma = UDU^T$ with some orthonormal matrix $U \in \mathcal{O}(p,r) = \{V \in \mathbb{R}^{p\times r} : V^TV = I_r\}$ and some diagonal matrix D whose diagonal entires are all positive. Then, the Moore-Penrose pseudo inverse of Γ is defined by

$$\Gamma^+ = UD^{-1}U^T.$$

Lemma 3.8.1. *Consider a symmetric and invertible matrix $R \in \mathbb{R}^{r\times r}$. Then, we have*

$$R = V^T(VR^{-1}V^T)^+V, \tag{3.71}$$

for any $V \in \mathcal{O}(p,r)$.

Proof. To prove (3.71), we write R as its eigenvalue decomposition $W\Lambda W^T$ for some $W \in \mathcal{O}(r,r)$ and some invertible diagonal matrix Λ. Then, it is easy to see that $VW \in \mathcal{O}(p,r)$, and we thus have $(VR^{-1}V^T)^+ = (VW\Lambda^{-1}W^TV^T)^+ = VW\Lambda W^TV^T$, and then $V^T(VR^{-1}V^T)^+V = V^TVW\Lambda W^TV^TV = W\Lambda W^T = R$. □

Lemma 3.8.2. *Let $A, B \in \mathbb{R}^{n\times m}$ be matrices of full column rank m (i.e. $m \le n$). Let Z_A and Z_B span the nullspaces of A and B, respectively. That is, $Z_A, Z_B \in \mathbb{R}^{n\times(n-m)}$ and*

$$A^TZ_A = 0, \quad B^TZ_B = 0.$$

Then, we have

$$I_n - A(B^TA)^{-1}B^T = Z_B(Z_A^TZ_B)^{-1}Z_A^T.$$

Proof. Let $C = I_n - A(B^TA)^{-1}B^T$, and then it is easy to check that

$$CZ_B = Z_B, \quad Z_A^TC = Z_A^T, \quad CA = 0, \quad B^TC = 0.$$

Note that the above four equations determine the singular value decomposition of C and are also satisfied by $Z_B(Z_A^TZ_B)^{-1}Z_A^T$, which immediately implies $C = Z_B(Z_A^TZ_B)^{-1}Z_A^T$. □

Lemma 3.8.3. *Suppose for symmetric matrices $S, H \in \mathbb{R}^{p\times p}$, we have $\mathcal{M}([S; H]) = \mathbb{R}^p$, where the notation $\mathcal{M}(\cdot)$ means the subspace spanned by the columns of a matrix. Then, we have*

$$(tS + H)^{-1} \to R(R^THR)^{-1}R^T,$$

as $t \to \infty$, where R is any matrix such that $\mathcal{M}(R)$ is the null space of S.

Proof. We first prove the special case of $H = I_p$. Denote the rank of S by r, and then S has an eigenvalue decomposition $S = UDU^T$ for some $U \in \mathcal{O}(p, r)$ and some diagonal matrix D with positive diagonal entries. Since $\mathcal{M}(R)$ is the null space of S, we have $I_p = UU^T + R(R^TR)^{-1}R^T$ by Lemma 3.8.2. Then,

$$(tS + I_p)^{-1} = (U(tD + I_r)U^T + R(R^TR)^{-1}R^T)^{-1} = U(tD + I_r)^{-1}U^T + R(R^TR)^{-1}R^T,$$

which converges to $R(R^TR)^{-1}R^T$ as $t \to \infty$. The second equality above follows from $U^TR = 0$. Now we assume a general H of full rank, which means $H = Q^TQ$ for some $Q \in \mathbb{R}^{p\times p}$ that is invertible. Then,

$$(tS + H)^{-1} = Q^{-1}(t(Q^T)^{-1}SQ^{-1} + I_p)^{-1}(Q^T)^{-1}.$$

Since the null space of $(Q^T)^{-1}SQ^{-1}$ is $\mathcal{M}(QR)$, we have

$$(t(Q^T)^{-1}SQ^{-1} + I_p)^{-1} \to QR(R^TQ^TQR)^{-1}R^TQ^T = QR(R^THR)^{-1}R^TQ^T,$$

and therefore $(tS+H)^{-1} \to R(R^THR)^{-1}R^T$. For a general H that is not necessarily full rank, since $\mathcal{M}([S;H]) = \mathbb{R}^p$, $S+H$ is a matrix of full rank. Then,

$$(tS+H)^{-1} = ((t-1)S+S+H)^{-1} \to R(R^T(S+H)R)^{-1}R^T = R(R^THR)^{-1}R^T,$$

and the proof is complete. □

3.8.2 Degenerate Gaussian Distributions

A multivariate Gaussian distribution is fully characterized by its mean vector and covariance matrix. For $N(\mu, \Sigma)$ with some $\mu \in \mathbb{R}^p$ and a positive semidefinite $\Sigma \in \mathbb{R}^{p\times p}$, we call the distribution degenerate if $\mathsf{rank}(\Sigma) < p$. Given any Σ such that $\mathsf{rank}(\Sigma) = r < p$, we have the decomposition $\Sigma = AA^T$ for some $A \in \mathbb{R}^{p\times r}$. Therefore, $X \sim N(\mu, \Sigma)$ if and only if

$$X = \mu + AZ, \tag{3.72}$$

where $Z \sim N(0, I_r)$. The latent variable representation (3.72) immediately implies that $X - \mu \in \mathcal{M}(A) = \mathcal{M}(\Sigma)$ with probability one.

The density function of $N(\mu, \Sigma)$ is given by

$$(2\pi)^{-r/2}\frac{1}{\sqrt{\det{}_+(\Sigma)}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^+(x-\mu)\right)\mathbb{I}\{x-\mu \in \mathcal{M}(\Sigma)\}. \tag{3.73}$$

The formula (3.73) can be found in [84, 140]. Note that the density function (3.73) is defined with respect to the Lebesgue measure on the subspace $\{x : x-\mu \in \mathcal{M}(\Sigma)\}$. Here, the $\det_+(\cdot)$ is used for the product of all nonzero eigenvalues of a symmetric matrix, and Σ^+ is the Moore-Penrose inverse of the covariance matrix Σ. The two characterizations (3.72) and (3.73) of $N(\mu, \Sigma)$ are equivalent to each other.

The property is useful for us to identify whether a formula leads to a well-defined density function of a degenerate Gaussian distribution.

Lemma 3.8.4. *Suppose* $f(x) = \exp(-\frac{1}{2}(x-\mu)^T\Omega(x-\mu))\mathbb{I}\{x-\mu \in \mathcal{M}(V)\}$ *for some* $\mu \in \mathbb{R}^p$, *some positive semidefinite* $\Omega \in \mathbb{R}^{p\times p}$ *and some* $V \in \mathcal{O}(p,r)$. *As long as* $\mathcal{M}(\Omega V) = \mathcal{M}(V)$, *we have* $\int f(x)dx < \infty$, *and* $f(x)/\int f(x)dx$ *is the density function of* $N(\mu,\Sigma)$ *with* $\Sigma = V(V^T\Omega V)^{-1}V^T$.

Proof. Without loss of generality, assume $\mu = 0$. Since $\mathcal{M}(\Omega V) = \mathcal{M}(V)$, $V^T\Omega V$ is an invertible matrix, and thus Σ is well defined. It is easy to see that $\mathcal{M}(V) = \mathcal{M}(\Sigma)$. Therefore, in view of (3.73), we only need to show

$$x^T\Omega x = x^T(V(V^T\Omega V)^{-1}V^T)^+x,$$

for all $x \in \mathcal{M}(V)$. Since $x = VV^Tx$ for all $x \in \mathcal{M}(V)$, it suffices to show

$$V^T\Omega V = V^T(V(V^T\Omega V)^{-1}V^T)^+V,$$

which is immediately implied by (3.71) with $R = V^T\Omega V$. The proof is complete. □

We remark that Lemma 3.8.4 also holds for a $V \in \mathbb{R}^{p\times r}$ that satisfies $\mathsf{rank}(V) = r$ but is not necessarily orthonormal. This is because $V(V^T\Omega V)^{-1}V^T = W(W^T\Omega W)^{-1}W^T$ whenever $\mathcal{M}(V) = \mathcal{M}(W)$.

3.8.3 Proofs of Propositions

Proof of Proposition 3.2.1. The property of the Laplacian matrix L_γ is standard in spectral graph theory [141]. We apply Lemma 3.8.4 with $\mu = 0$, $\Omega = \sigma^{-2}L_\gamma$ and V is chosen arbitrarily from $\mathcal{O}(p,p-1)$ such that $w^TV = 0$. Then, the condition $\mathcal{M}(\Omega V) = \mathcal{M}(V)$ is equivalent to $\mathbb{1}_p^Tw \neq 0$, because $\mathbb{1}_p$ spans the null space of L_γ. Note that (3.71) immediately implies $VRV^T =$

$VV^T(VR^{-1}V^T)^+VV^T = (VR^{-1}V^T)^+$. We then have

$$\begin{aligned}
\det{}_+(V(V^T\Omega V)^{-1}V^T) &= \sigma^{2(p-1)}\det{}_+(V(V^TL_\gamma V)^{-1}V^T) \\
&= \sigma^{2(p-1)}\det{}_+((VV^TL_\gamma VV^T)^+) \\
&= \sigma^{2(p-1)}\frac{1}{\det{}_+(VV^TL_\gamma VV^T)}.
\end{aligned}$$

The proof is complete by realizing that $VV^T = I_p - ww^T/\|w\|^2$ from Lemma 3.8.2. □

Proof of Proposition 3.3.1. Recall the incidence matrix $D \in \mathbb{R}^{m\times p}$ defined in subsection 3.2.1. With the new notations $\Phi = D^T\,\mathrm{diag}(\gamma)D$ and $\Psi = D^T\,\mathrm{diag}(1-\gamma)D$, we can write $L_\gamma = v_0^{-1}\Phi + v_1^{-1}\Psi$ and $\widetilde{L}_\gamma = v_1^{-1}\Psi$.

We first prove that (3.23) is well defined. By Lemma 3.8.4, it is sufficient to show $\widetilde{V}^TZ_\gamma^T\Psi Z_\gamma\widetilde{V}$ is invertible. This is because $\mathcal{M}([\Phi;\Psi;w]) = \mathbb{R}^p$, and the columns of $Z_\gamma\widetilde{V}$ are all orthogonal to $\mathcal{M}([\Phi;w])$. Therefore, (3.23) is well defined and its covariance matrix is given by (3.74).

Now we will show the distribution (3.8) converges to that of $Z_\gamma\widetilde{\theta}$ with $\widetilde{\theta}$ distributed by (3.23) as $v_0 \to 0$. Let $\widetilde{V} \in \mathbb{R}^{r\times r-1}$ be a matrix of rank $r-1$ that satisfies $\widetilde{V}^TZ_\gamma^Tw = 0$. Then, by Lemma 3.8.4, the distribution (3.23) can be written as

$$\widetilde{\theta} \sim N(0, \sigma^2\widetilde{V}(\widetilde{V}^TZ_\gamma^T\widetilde{L}_\gamma Z_\gamma\widetilde{V})^{-1}\widetilde{V}^T), \tag{3.74}$$

which implies

$$Z_\gamma\widetilde{\theta} \sim N(0, \sigma^2Z_\gamma\widetilde{V}(\widetilde{V}^TZ_\gamma^T\widetilde{L}_\gamma Z_\gamma\widetilde{V})^{-1}\widetilde{V}^TZ_\gamma^T).$$

On the other hand, the distribution (3.8) can be written as

$$\theta \sim N(0, \sigma^2V(V^TL_\gamma V)^{-1}V^T),$$

where the matrix $V \in \mathbb{R}^{p\times p-1}$ can be chosen to be any matrix of rank $p-1$ that satisfies $V^Tw = 0$.

In particular, we can choose V that takes the form of

$$V = [Z_\gamma \widetilde{V}; U],$$

where $U \in \mathcal{O}(p, p-r)$ and U satisfies $U^T w = 0$ and $U^T Z_\gamma \widetilde{V} = 0$. Since both θ and $Z_\gamma \widetilde{\theta}$ are Gaussian random vectors, we need to prove

$$V(V^T L_\gamma V)^{-1} V^T \to Z_\gamma \widetilde{V} (\widetilde{V}^T Z_\gamma^T \widetilde{L}_\gamma Z_\gamma \widetilde{V})^{-1} \widetilde{V}^T Z_\gamma^T, \tag{3.75}$$

as $v_0 \to 0$. Note that (3.75) is equivalent to

$$V(v_0^{-1} V^T \Phi V + v_1^{-1} V^T \Psi V)^{-1} V^T \to v_1 Z_\gamma \widetilde{V} (\widetilde{V}^T Z_\gamma^T \Psi Z_\gamma \widetilde{V})^{-1} \widetilde{V}^T Z_\gamma^T, \tag{3.76}$$

as $v_0 \to 0$. By the definition of Z_γ and the property of graph Laplacian, the null space of Φ is $\mathcal{M}(Z_\gamma)$. By the construction of V, the null space of $V^T \Phi V$ is $\mathcal{M}(V^T Z_\gamma)$. Moreover, we have $\mathcal{M}([V^T \Phi V; V^T \Psi V]) = \mathbb{R}^{p-1}$. Therefore, Lemma 3.8.3 implies that

$$(v_0^{-1} V^T \Phi V + v_1^{-1} V^T \Psi V)^{-1} \to v_1 V^T Z_\gamma (Z_\gamma^T V V^T \Psi V V^T Z_\gamma)^{-1} Z_\gamma^T V,$$

and thus

$$V(v_0^{-1} V^T \Phi V + v_1^{-1} V^T \Psi V)^{-1} V^T \to v_1 V V^T Z_\gamma (Z_\gamma^T V V^T \Psi V V^T Z_\gamma)^{-1} Z_\gamma^T V V^T.$$

Since $V V^T Z_\gamma = Z_\gamma \widetilde{V} \widetilde{V}^T Z_\gamma^T Z_\gamma$, we have $\mathcal{M}(V V^T Z_\gamma) = \mathcal{M}(Z_\gamma \widetilde{V})$. This implies

$$V V^T Z_\gamma (Z_\gamma^T V V^T \Psi V V^T Z_\gamma)^{-1} Z_\gamma^T V V^T = Z_\gamma \widetilde{V} (\widetilde{V}^T Z_\gamma^T \Psi Z_\gamma \widetilde{V})^{-1} \widetilde{V}^T Z_\gamma^T,$$

and therefore we obtain (3.76). The proof is complete. □

Proof of Proposition 3.4.1. We only prove the case with $d = 1$. The general case with $d \geq 2$ follows

the same argument with more complicated notation of covariance matrices (such as Kronecker products). By Lemma 3.8.4, (3.33) is the density function of $N(0, \Sigma)$, with

$$\Sigma = \sigma^2 U (U^T L_\gamma U)^{-1} U^T,$$

where

$$L_\gamma = \begin{bmatrix} (v_0^{-1} - v_1^{-1}) I_n + v_1^{-1} k I_n & -(v_0^{-1} - v_1^{-1})\gamma - v_1^{-1} \mathbb{1}_{n \times k} \\ -(v_0^{-1} - v_1^{-1})\gamma^T - v_1^{-1} \mathbb{1}_{k \times n} & (v_0^{-1} - v_1^{-1})\gamma^T \gamma + v_1^{-1} n I_k \end{bmatrix}.$$

The matrix U is defined by

$$U = \begin{bmatrix} V & 0_{n \times k} \\ 0_{k \times (n-1)} & I_k \end{bmatrix},$$

and $V \in \mathbb{R}^{n \times (n-1)}$ is a matrix of rank $n-1$ that satisfies $\mathbb{1}_n^T V = 0$. Note that $\theta | \gamma, \sigma^2$ follows $N(0, \Sigma_{[n] \times [n]})$. That is, the covariance matrix is the top $n \times n$ submatrix of Σ. A direct calculation gives

$$\Sigma_{[n] \times [n]} = \sigma^2 V [V^T (A - B C^{-1} B^T) V]^{-1} V^T,$$

where

$$\begin{aligned} A &= (v_0^{-1} - v_1^{-1}) I_n + v_1^{-1} k I_n, \\ B &= -(v_0^{-1} - v_1^{-1})\gamma - v_1^{-1} \mathbb{1}_{n \times k}, \\ C &= (v_0^{-1} - v_1^{-1})\gamma^T \gamma + v_1^{-1} n I_k. \end{aligned}$$

Letting $v_1 \to \infty$, we have

$$\Sigma_{[n] \times [n]} \to \sigma^2 v_0 V [V^T (I_n - \gamma (\gamma^T \gamma)^{-1} \gamma^T) V]^{-1} V^T.$$

The existence of $(\gamma^T \gamma)^T$ is guaranteed by the condition that γ is non-degenerate. By Lemma 3.8.4, $p(\theta | \gamma, \sigma^2) \propto \prod_{1 \le i < l \le n} \exp\left(-\frac{\lambda_{il} \|\theta_i - \theta_l\|^2}{2\sigma^2 v_0} \right) \mathbb{I}\{\mathbb{1}_n^T \theta = 0\}$ is the density function of

$N(0, \sigma^2 v_0 V[V^T(I_n - \gamma(\gamma^T\gamma)^{-1}\gamma^T)V]^{-1}V^T)$, which completes the proof. □

Proof of Proposition 3.4.2. We only prove the case with $d = 1$. The general case with $d \geq 2$ follows the same argument with more complicated notation of covariance matrices (such as Kronecker products). The proof is basically an application of Proposition 3.3.1. That is, as $v_0 \to 0$, the distribution of $(\theta^T, \mu^T)^T$ weakly converges to that of $Z_\gamma \widetilde{\mu}$. In the current setting, we have

$$Z_\gamma = \begin{bmatrix} \gamma \\ I_k \end{bmatrix}.$$

The random vector $\widetilde{\mu}$ is distributed by (3.24). Note that the contracted base graph is a complete graph on $\{1, ..., k\}$, and $w_{jl} = n_j + n_l$ in the current setting. The density (3.24) thus becomes

$$p(\widetilde{\mu}|\gamma, \sigma^2) \propto \prod_{1 \leq j < l \leq k} \exp\left(-\frac{(n_j + n_l)(\widetilde{\mu}_j - \widetilde{\mu}_l)^2}{2\sigma^2 v_1}\right) \mathbb{I}\{\mathbb{1}_n^T \gamma \widetilde{\mu} = 0\}.$$

Finally, the relations $\theta = \gamma\widetilde{\mu}$ and $\mu = \widetilde{\mu}$ lead to the desired conclusion. □

Proof of Proposition 3.6.1. Note that the integration is with respect to the Lebesgue measure on the $(n-1)$-dimensional subspace $\{\theta : \mathbb{1}_n^T\theta = 0\}$. Consider a matrix $V \in \mathbb{R}^{n \times n-1}$ of rank $n-1$ that satisfies $\mathbb{1}_n^T V = 0$, which means that the columns of $[\mathbb{1}_n : V] \in \mathbb{R}^{n \times n}$ form a nondegenerate basis. Then, we can write $d\theta$ in the integral as $\frac{1}{\sqrt{\det(V^T V)}} d(V^T\theta)$. For the Laplacian matrix L_γ that satisfies $\theta^T L_\gamma \theta = \sum_{i=1}^{n-1} \frac{(\theta_{i+1} - \theta_i)^2}{v_0\gamma_i + v_1(1-\gamma_i)}$, we have $\mathbb{1}_n^T L_\gamma = 0$. In particular, we choose V such that its ith column is $V_{*i} = e_i - e_{i+1}$, where e_i is a vector whose ith entry is 1 and 0 elsewhere.

Then, we have $L_\gamma = V S_\gamma V^T$ with $S_\gamma = \mathrm{diag}(v_0^{-1}\gamma + v_1^{-1}(1-\gamma))$, and the integral becomes

$$
\begin{aligned}
& \int_{\mathbb{1}_n^T\theta=0, \theta_1\le\ldots\le\theta_n} 2^{(n-1)} \frac{1}{(2\pi\sigma^2)^{(n-1)/2}} \sqrt{\det{}_{\mathbb{1}_n}(L_\gamma)} \exp\left(-\frac{1}{2\sigma^2}\theta^T L_\gamma \theta\right) d\theta \\
= & \int_{\theta_1\le\ldots\le\theta_n} 2^{(n-1)} \frac{1}{(2\pi\sigma^2)^{(n-1)/2}} \sqrt{\frac{\det{}_{\mathbb{1}_n}(L_\gamma)}{\det(V^TV)}} \exp\left(-\frac{1}{2\sigma^2}\theta^T V S_\gamma V^T \theta\right) d(V^T\theta) \\
= & \int_{\delta_1\le 0,\ldots,\delta_n\le 0} 2^{(n-1)} \frac{1}{(2\pi\sigma^2)^{(n-1)/2}} \sqrt{\frac{\det{}_{\mathbb{1}_n}(L_\gamma)}{\det(V^TV)}} \exp\left(-\frac{1}{2\sigma^2}\delta^T S_\gamma \delta\right) d\delta \\
= & \int \frac{1}{(2\pi\sigma^2)^{(n-1)/2}} \sqrt{\frac{\det{}_{\mathbb{1}_n}(L_\gamma)}{\det(V^TV)}} \exp\left(-\frac{1}{2\sigma^2}\delta^T S_\gamma \delta\right) d\delta \qquad (3.77) \\
= & \sqrt{\frac{\det{}_{\mathbb{1}_n}(L_\gamma)}{\det(S_\gamma)\det(V^TV)}} \qquad (3.78) \\
= & \ 1,
\end{aligned}
$$

where the last equality is by $\det_{\mathbb{1}_n}(L_\gamma) = \det_+(V S_\gamma V^T) = \det(S_\gamma)\det(V^TV)$. The equality (3.77) is by the symmetry of $\exp\left(-\frac{1}{2\sigma^2}\delta^T S_\gamma \delta\right)$, and (3.78) is by Lemma 3.8.4. Finally, Lemma 3.3.1 says that

$$\det{}_{\mathbb{1}_n}(L_\gamma) = n \prod_{i=1}^{n-1} [v_0^{-1}\gamma_i + v_1^{-1}(1-\gamma_i)]$$

This completes the proof. □

Proof of Proposition 3.6.2. We need to calculate

$$\int_{\mathbb{1}_n^T Z_\gamma \widetilde{\theta}=0,\ \widetilde{\theta}_1\le\cdots\le\widetilde{\theta}_s} \prod_{l=1}^{s} \exp\left(-\frac{(\widetilde{\theta}_l - \widetilde{\theta}_{l+1})^2}{2\sigma^2 v_1}\right) d\widetilde{\theta},$$

where the integral is taken with respect to the Lebesgue measure on the low-dimensional subspace $\{\widetilde{\theta} : \mathbb{1}_n^T Z_\gamma \widetilde{\theta} = 0\}$. Choose $\widetilde{V}_{*l} = e_l - e_{l+1}$, where $e_l \in \{0,1\}^s$ is a vector whose lth entry is 1 and 0 elsewhere. Then the columns of $[Z_\gamma^T \mathbb{1}_n : (Z_\gamma^T Z_\gamma)^{-1}\widetilde{V}] \in \mathbb{R}^{s\times s}$ form a non-degenerate basis of $\mathbb{R}^s$. This is because

$$Z_\gamma^T \mathbb{1}_n = (n_1, \cdots, n_s)^T, \quad Z_\gamma^T Z_\gamma = \mathrm{diag}(n_1, \cdots, n_s),$$

where n_l is the size of lth cluster. Furthermore, $Z_\gamma^T \mathbb{1}_n$ and $(Z_\gamma^T Z_\gamma)^{-1}\widetilde{V}$ are orthogonal to each other. We write $\widetilde{W} = (Z_\gamma^T Z_\gamma)^{-1}\widetilde{V}$ for simplicity. Then,

$$
\begin{aligned}
& \int_{\mathbb{1}_n^T Z_\gamma \widetilde{\theta}=0,\, \widetilde{\theta}_1 \le \cdots \le \widetilde{\theta}_s} \prod_{l=1}^{s} \exp\left(-\frac{(\widetilde{\theta}_l - \widetilde{\theta}_{l+1})^2}{2\sigma^2 v_1}\right) d\widetilde{\theta} \\
= & \frac{1}{\sqrt{\det \widetilde{W}^T \widetilde{W}}} \int_{\widetilde{\theta}_1 \le \cdots \le \widetilde{\theta}_s} \prod_{l=1}^{s} \exp\left(-\frac{(\widetilde{\theta}_l - \widetilde{\theta}_{l+1})^2}{2\sigma^2 v_1}\right) d(\widetilde{W}^T \widetilde{\theta}) \\
= & \frac{\det \widetilde{W}^T \widetilde{W} (\widetilde{V}^T \widetilde{W})^{-1}}{\sqrt{\det \widetilde{W}^T \widetilde{W}}} \int_{\widetilde{\theta}_1 \le \cdots \le \widetilde{\theta}_s} \prod_{l=1}^{s-1} \exp\left(-\frac{(\widetilde{\theta}_l - \widetilde{\theta}_{l+1})^2}{2\sigma^2 v_1}\right) d(\widetilde{V}^T \widetilde{\theta}) \\
= & \frac{\det \widetilde{W}^T \widetilde{W} (\widetilde{V}^T \widetilde{W})^{-1}}{\sqrt{\det \widetilde{W}^T \widetilde{W}}} \int_{\widetilde{\delta}_1, \cdots, \widetilde{\delta}_{s-1} \le 0} \prod_{l=1}^{s-1} \exp\left(-\frac{\widetilde{\delta}_l^2}{2\sigma^2 v_1}\right) d\widetilde{\delta} \\
= & \frac{\sqrt{\det \widetilde{W}^T \widetilde{W}}}{\det \widetilde{V}^T \widetilde{W}} \int_{\widetilde{\delta}_1, \cdots, \widetilde{\delta}_{s-1} \le 0} \prod_{l=1}^{s-1} \exp\left(-\frac{\widetilde{\delta}_l^2}{2\sigma^2 v_1}\right) d\widetilde{\delta} \\
= & \frac{(2\pi\sigma^2 v_1)^{(s-1)/2}}{2^{s-1}} \times \frac{\sqrt{\det \widetilde{W}^T \widetilde{W}}}{\det \widetilde{V}^T \widetilde{W}}.
\end{aligned}
$$

The first equality is from the Lebesgue integration on the reduced space and the last equality is by symmetry. The second equality follows from the change of variables formula, because for any $\widetilde{\theta}$ such that $\mathbb{1}_n^T Z_\gamma \widetilde{\theta} = 0$, $\widetilde{\theta} = \widetilde{W} U$ for some U, which leads to $\widetilde{W}^T \widetilde{W} (\widetilde{V}^T \widetilde{W})^{-1} \widetilde{V}^T \widetilde{\theta} = \widetilde{W}^T \widetilde{W} U = \widetilde{W}^T \widetilde{\theta}$. Finally, we observe that

$$
\begin{aligned}
v_1^{(s-1)/2} \sqrt{\det{}_{Z_\gamma^T \mathbb{1}_n}(Z_\gamma^T \widetilde{L}_\gamma Z_\gamma)} &= \left(\frac{\det(\widetilde{W}^T \widetilde{V} \widetilde{V}^T \widetilde{W})}{\det \widetilde{W}^T \widetilde{W}}\right)^{1/2} \\
&= \left(\frac{(\det \widetilde{V}^T \widetilde{W})^2}{\det \widetilde{W}^T \widetilde{W}}\right)^{1/2} = \frac{\det \widetilde{V}^T \widetilde{W}}{\sqrt{\det \widetilde{W}^T \widetilde{W}}},
\end{aligned}
$$

since $\widetilde{V}\widetilde{V}^T = v_1 \widetilde{L}_\gamma$ is the reduced graph Laplacian and the columns of $\widetilde{W}$ spans the nullspace of $Z_\gamma^T \mathbb{1}_n$. The proof is complete. □

3.8.4 *Proof of Lemma 3.3.1*

We let $U \in \mathcal{O}(p, p-1)$ be an orthonormal matrix that satisfies $\mathbb{1}_p^T U = 0$, and $V \in \mathcal{O}(p, p-1)$ be an orthonormal matrix that satisfies $w^T V = 0$. Write $I_p - U(V^T U)^{-1} V^T$ as R. Then, $RU = 0$, which implies that R has rank at most one. The facts $Rw = w$ and $\mathbb{1}_p^T R = \mathbb{1}_p^T$, together with Lemma 3.8.2, imply that

$$I_p - U(V^T U)^{-1} V^T = \frac{1}{\mathbb{1}_p^T w} w \mathbb{1}_p^T. \tag{3.79}$$

Therefore,

$$\begin{aligned} \det{}_w(L_\gamma) &= \det{}_+(V V^T L_\gamma V V^T) \\ &= \det(V^T L_\gamma V) \\ &= \det(V^T U U^T L_\gamma U U^T V) \qquad (3.80) \\ &= (\det(V^T U))^2 \det(U^T L_\gamma U). \end{aligned}$$

The inequality (3.80) is because UU^T is a projection matrix to the null space of L_γ. We are going to calculate $\det(V^T U)$ and $\det(U^T L_\gamma U)$ separately. For $\det(V^T U)$, we have

$$\begin{aligned} 1 &= \det\left(\begin{bmatrix} V & \|w\|^{-1} w \end{bmatrix}^T \begin{bmatrix} U & p^{-1/2} \mathbb{1}_p \end{bmatrix} \right) \\ &= \det\left(\begin{bmatrix} V^T U & p^{-1/2} V^T \mathbb{1}_p \\ \|w\|^{-1} w^T U & w^T \mathbb{1}_p / (p^{1/2} \|w\|) \end{bmatrix} \right) \\ &= \det(V^T U) \det(w^T (I_p - U(V^T U)^{-1} V^T) \mathbb{1}_p) / (p^{1/2} \|w\|) \\ &= \det(V^T U) \frac{p^{1/2} \|w\|}{\mathbb{1}_p^T w}, \end{aligned}$$

and we thus get

$$\Big(\det(V^T U)\Big)^2 = \frac{\Big(\mathbb{1}_p^T w\Big)^2}{p \|w\|^2}. \tag{3.81}$$

We use (3.79) for the equality (3.81).

The calculation of $\det(U^T L_\gamma U)$ requires the Cauchy-Binet formula. For any given matrices $A, B \in \mathbb{R}^{n\times m}$ with $n \le m$, we have

$$\det(AB^T) = \sum_{\{S\subset[m]:|S|=n\}} \det(B_{*S}^T A_{*S}). \tag{3.82}$$

The version (3.82) can be found in [150]. Let $A = B = U^T D^T (v_0^{-1/2}\,\mathrm{diag}(\gamma) + v_1^{-1/2}\,\mathrm{diag}(1-\gamma))$, and we have

$$\det(U^T L_\gamma U) = \sum_{\{S\subset E:|S|=p-1\}} \left(\prod_{(i,j)\in S} \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right] \right) \det(D_{S*} U U^T D_{S*}^T).$$

Note that $\det(D_{S*}UU^T D_{S*}^T) = \det(U^T D_{S*}^T D_{S*} U)$, and $D_{S*}^T D_{S*}$ is the graph Laplacian of a subgraph of the base graph with the edge set S. Since $|S| = p-1$, S is either a spanning tree ($\det(U^T D_{S*}^T D_{S*} U) = p$) or is disconnected ($\det(U^T D_{S*}^T D_{S*} U) = 0$), we have

$$\det(U^T L_\gamma U) = p \sum_{T\in \mathsf{spt}(G)} \prod_{(i,j)\in T} \left[v_0^{-1}\gamma_{ij} + v_1^{-1}(1-\gamma_{ij}) \right]. \tag{3.83}$$

Therefore, by plugging (3.81) and (3.83) into (3.80), we obtain the desired conclusion.

3.8.5 Some Implementation Details

In this subsection, we present a fast algorithm that solves the M-step (3.57). To simplify the notation in the discussion, we consider a special case with $X_1 = I_{n_1}$, $X_2 = I_{n_2}$, $w = \mathbb{1}_{n_1}\mathbb{1}_{n_2}^T$ and $\nu = 0$, which is the most important setting that we need for the biclustering problem. In other words, we need to optimize $F(\theta; q_1, q_2)$ over $\theta \in \Theta_w$ for any q_1 and q_2, where

$$\begin{aligned} F(\theta; q_1, q_2) &= \|y - \bar{y}\mathbb{1}_{n_1}\mathbb{1}_{n_2}^T - \theta\|_{\mathrm{F}}^2 + \mathsf{vec}(\theta)^T \left(L_{q_2} \otimes I_{p_1} + I_{p_2} \otimes L_{q_1} \right) \mathsf{vec}(\theta) \quad (3.84) \\ &= \|y - \bar{y}\mathbb{1}_{n_1}\mathbb{1}_{n_2}^T - \theta\|_{\mathrm{F}}^2 + \left\langle \theta\theta^T, L_{q_1} \right\rangle + \left\langle \theta^T\theta, L_{q_2} \right\rangle. \quad (3.85) \end{aligned}$$

Algorithm 5 A fast DLPA

Input: Initialize u_1, u_2 and z_2.

repeat

$z_1^{new} = (I_{n_1} + L_{q_1})^{-1}(z_2 + u_2)$

$z_2^{new} = (z_1^{new} + u_1)(I_{n_2} + L_{q_2})^{-1}$

$u_1^{new} = z_2 + u_2 - z_1^{new},\ u_2^{new} = z_1^{new} + u_2 - z_2^{new}$

$z_1 = z_1^{new}, z_2 = z_2^{new}, u_1 = u_1^{new}, u_2 = u_2^{new}$

until convergence criteria met

Output: $\theta = z_2$

Algorithm 5 is a Dykstra-like proximal algorithm (DLPA) [40] that iteratively solves the optimization problem. It is shown that Algorithm 5 has a provable linear convergence [31]. If we initialize $u_1 = u_2 = 0_{n_1 \times n_2}$ and $z_2 = y - \bar{y}\mathbb{1}_{n_1}\mathbb{1}_{n_2}^T$ with $\hat{\alpha} = \bar{y}$, then the first two steps of Algorithm 5 can be written as the following update

$$\theta^{\text{new}} = (I_{n_1} + L_{q_1})^{-1} \left(y - \bar{y}\mathbb{1}_{n_1}\mathbb{1}_{n_2}^T \right) (I_{n_2} + L_{q_2})^{-1}. \tag{3.86}$$

In practice, we suggest using (3.86) as approximate M-step updates in the first few iterations of the EM algorithm. Then, the full version of Algorithm 5 can be implemented in later iterations to ensure convergence.

Table 3.7: A List of Graphs Used for Evaluating Quality of Variational Approximation

Tadpole(50,50)	Lollipop(80,20)	Grid(10,10)	Grid(20,5)	Bipartite(80,20)	Barbell(50,50)
$P_{50} \vee C_{50}$	$P_{80} \vee K_{20}$	$P_{10} \otimes P_{10}$	$P_{20} \otimes P_5$	$K_{80,20}$	$K_{50} \vee K_{50}$

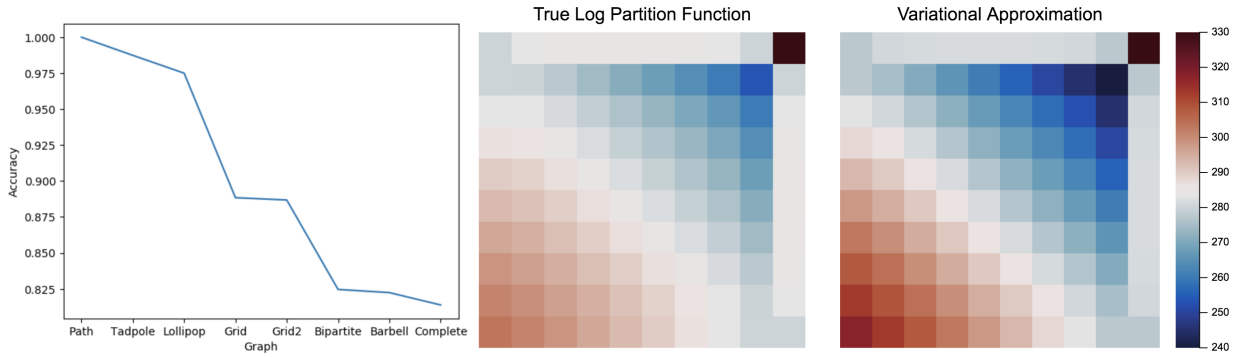


Figure 3.19: (Left) Accuracy of variational approximation to the normalization constant; (Center and Right) The quality of variational approximation on Grid(10,10).

3.8.6 Accuracy of Variational Approximation

In this subsection, we conduct an empirical study of the accuracy of the variational approximation. The variational lower bound we used is

$$\begin{aligned}
&\log \sum_{T \in \text{spt}(G)} \prod_{(i,j) \in T} \left[v_0^{-1} \gamma_{ij} + v_1^{-1} (1 - \gamma_{ij}) \right] \\
&\geq \sum_{(i,j) \in E} r_{ij} \log \left[v_0^{-1} \gamma_{ij} + v_1^{-1} (1 - \gamma_{ij}) \right] + \log |\text{spt}(G)|,
\end{aligned}$$

where $r_{ij} = |\text{spt}(G)|^{-1} \sum_{T \in \text{spt}(G)} \mathbb{I}\{(i,j) \in T\}$ is an effective resistance of an edge (i,j). An effective resistance r_e of an edge e measures how important the edge e is in the whole graph, provided that their local resistance is 1. For instance, if (i,j) is the only edge connecting two mutually exclusive subgraphs containing i and j respectively then $r_{ij} = 1$. In this case, determining whether $\gamma_{ij} = 1$ has a direct effect on separation of i and j, one may want to put larger weights on γ_{ij} in the objective function. In the right hand side of (3.19), $\log[v_0^{-1} \gamma_{ij} + v_1^{-1}(1 - \gamma_{ij})]$ is weighted by the effective resistance r_{ij} of the edge (i,j). This implies that the variational lower bound or the right hand side of the above inequality puts larger weights to more "important" edges. This intuitive explanation can be verified by the following numerical experiments.

We compare the true log partition function

$$f_1(\gamma) = \log \sum_{T \in \text{spt}(G)} \prod_{(i,j) \in T} \left[v_0^{-1} \gamma_{ij} + v_1^{-1} (1 - \gamma_{ij}) \right]$$

with our variational lower bound

$$f_2(\gamma) = \sum_{(i,j) \in E} r_{ij} \log \left[v_0^{-1} \gamma_{ij} + v_1^{-1} (1 - \gamma_{ij}) \right] + \log |\text{spt}(G)|$$

for various choices of graphs. To this end, we randomly sample two weighted graphs G_1 and G_2 and then put independent random uniform weights on the edges of G_1 and G_2, respectively. Then we compare the differences between the true normalization constants and the differences between the variational lower bounds. We fix $n = 100$, $v_0 = 10^{-1}$ and $v_1 = 10^3$. Let us write the linear chain graph as P_n and the complete graph as K_n. Let $\vee$ be a graph operator joining two graphs by adding exactly one connecting edge between two graphs. We compare P_n, K_n and the 6 graphs in Table 3.7. The graph $K_{80,20}$ is sampled under the additional constraint that $\sum_j \gamma_{ij} = 1$.

The left panel of Figure 3.19 plots the accuracy of approximation, measured by $e^{f_2(\gamma)}/e^{f_1(\gamma)}$ against the complexity of graphs. It shows that the approximation is more accurate for a sparser graph, but even for the complete graph we still obtain an accuracy over 0.8.

The center and the right panels of Figure 3.19 displays the true log-partition function and the variational approximation when Grid(10,10) is clustered into two parts. To be specific, the true signal $\theta^* \in \mathbb{R}^{10 \times 10}$ is defined on the nodes of G and has exactly 2 clusters, the bottom left square of (a, b) and the rest. That is,

$$\mathcal{C}_1^{(a,b)} = \{\theta_{ij} : i \leq a \text{ and } j \leq b\} \quad \mathcal{C}_2^{(a,b)} = \{\theta_{ij} : i > a \text{ or } j > b\}.$$

For instance, if $a = 3$ and $b = 4$, then $\mathcal{C}_1$ has 12 nodes. Let $\gamma^{(a,b)}$ be the corresponding latent structure parameter, and $\gamma_{ij}^{(a,b)} = 1$ if i and j are in the same cluster and $\gamma_{ij}^{(a,b)} = 0$ otherwise. For $a = 1, \cdots, 10$ and $b = 1, \cdots, 10$, we compare the true log-normalization constant $f_1(\gamma^{(a,b)})$ and

its variational approximation $f_2(\gamma^{(a,b)})$, and summarize the result in the center and the right panels of Figure 3.19. The result is displayed by heat-maps, and the values of $f_1(\gamma^{a,b})$ and $f_2(\gamma^{a,b})$ are reported in the (a, b)-th entries of the two heatmaps. The results support our conclusion that the variational approximations are reasonable for various graphical structures.

CHAPTER 4

A FAST ALGORITHM FOR MAXIMUM LIKELIHOOD ESTIMATION OF MIXTURE PROPORTIONS

4.1 Introduction

We consider maximum likelihood estimation of the mixture proportions in a finite mixture model where the component densities are known. The simplest example of this arises when we have independent and identically distributed (*i.i.d.*) observations $z_1, \ldots, z_n$ drawn from a finite mixture distribution with density

$$p(\,\cdot\,|\,x) = \sum_{k=1}^{m} x_k g_k(\,\cdot\,),$$

where the component densities $g_k(\cdot)$ are known and $x = (x_1, \ldots, x_m)^T$ denotes the unknown mixture proportions (non-negative and sum to one). Finding the maximum likelihood estimate (MLE) of x can be formulated as a constrained optimization problem:

$$\begin{array}{ll} \text{minimize} & f(x) \triangleq -\dfrac{1}{n}\displaystyle\sum_{j=1}^{n} \log\left(\sum_{k=1}^{m} L_{jk}x_k\right) \\ \text{subject to} & x \in \mathcal{S}^m \triangleq \left\{x : \sum_{k=1}^{m} x_k = 1,\ x \succeq 0\right\}, \end{array} \tag{4.1}$$

where L is an $n \times m$ matrix with entries $L_{jk} \triangleq g_k(z_j) \geq 0$. This optimization problem arises in many settings—including nonparametric empirical Bayes (EB) computations described later—where observations are not necessarily identically distributed. Here, we develop general methods for solving (4.1).

Problem (4.1) is a convex optimization problem and can be solved simply using expectation maximization (EM) [33]; see Supplementary Material. However, the convergence of EM can be intolerably slow [124, 5, 132, 158]; this slow convergence is illustrated evocatively in [91]. [91] and [90] show that modern convex optimization methods can be substantially faster and more reliable

3. This work is in collaboration with Peter Carbonetto, Matthew Stephens and Mihai Anitescu [87].

than EM. They demonstrate this by using an interior (IP) method to solve a dual formulation of the original problem. This method is implemented in the `KWDual` function of the R package `REBayes` [90], which interfaces to the commercial interior point solver `MOSEK` [3].

In this chapter, we provide an even faster algorithm for this problem based on sequential quadratic programming (SQP) [115]. The computational gains are greatest in large data sets where the matrix $L \in \mathbb{R}^{n \times m}$ is numerically rank-deficient. Rank deficiency can make the optimization problem harder to solve, even if it is convex [168]. As we show, a numerically rank-deficient L often occurs in the nonparametric EB problems that are the primary focus of `KWDual`. As an example of target problem size, we consider data from a genome-wide association study with $n > 10^6$ and $m > 100$. For such data, our methods are approximately 100 times faster than `KWDual` (about 10 seconds vs. 1,000 seconds). All our methods and numerical experiments are implemented in the Julia programming language [13], and the source code is available online at `https://github.com/stephenslab/mixsqp-paper`. Many of our methods are also implemented in an R package, `mixsqp`, which is available on CRAN [122].

4.2 Motivation: Nonparametric Empirical Bayes

Estimation of mixture proportions is a fundamental problem in statistics, dating back to at least [119]. This problem, combined with the need to fit increasingly large data sets, already provides strong motivation for finding efficient scalable algorithms for solving (4.1). Additionally, we are motivated by recent work on nonparametric approaches to empirical Bayes (EB) estimation in which a finite mixture with a large number of components is used to accurately approximate *nonparametric* families of prior distributions [91, 143]. Here we briefly discuss this motivating application.

We first consider a simple EB approach to solving the "normal means," or "Gaussian sequence," problem [81]. For $j = 1, \ldots, n$, we observe data z_j that are noisy observations of some underlying "true" values, θ_j, with normally distributed errors of known variance s_j^2,

$$z_j \,|\, \theta_j \sim N(\theta_j, s_j^2). \tag{4.2}$$

The EB approach to this problem assumes that θ_j are *i.i.d.* from some unknown distribution g,

$$\theta_j \,|\, g \sim g, \quad g \in \mathcal{G}, \tag{4.3}$$

where $\mathcal{G}$ is some specified class of distributions. The EB approach estimates g by maximizing the (marginal) log-likelihood, which is equivalent to solving:

$$\underset{g \in \mathcal{G}}{\text{minimize}} \; -\frac{1}{n} \sum_{j=1}^{n} \log \left[\int N(z_j; \theta, s_j^2)\, g(d\theta) \right], \tag{4.4}$$

where $N(\,\cdot\,; \mu, \sigma^2)$ denotes the normal density with mean μ and variance σ^2. After estimating g by solving (4.4), posterior statistics are computed for each j. Our focus is on the maximization step.

Although one can use simple parametric families for $\mathcal{G}$, in many settings one might prefer to use a more flexible nonparametric family. Examples include:

- $\mathcal{G} = \mathcal{R}$, the set of all real-valued distributions.
- $\mathcal{G} = \mathcal{U}_0$, the set of unimodal distributions with a mode at zero. (Extensions to a nonzero mode are straightforward.)
- $\mathcal{G} = \mathcal{SU}_0$, the set of symmetric unimodal distributions with a mode at zero.
- $\mathcal{G} = \mathcal{SN}_0$, the set of distributions that are scale mixtures of zero-mean normals, which includes several commonly used distributions, such as the t and double-exponential (Laplace) distributions.

The fully nonparametric case, $\mathcal{G} = \mathcal{R}$, is well studied (e.g., Laird 93, Jiang and Zhang 79, Brown and Greenshtein 24, Koenker and Mizera 91) and it is related to the classic Kiefer-Wolfowitz problem [85]. More constrained examples $\mathcal{G} = \mathcal{U}_0, \mathcal{SU}_0, \mathcal{SN}_0$ appear in [143] (see also Cordy and Thomas 32), and can be motivated by the desire to shrink estimates towards zero, or to impose some regularity on g without making strong parametric assumptions. For other motivating examples, see the nonparametric approaches to the "compound decision problem" described in [79] and [91].

The connection with (4.1) is that these nonparametric sets can be accurately approximated by a finite mixture with sufficiently large number of components; that is, they can be approximated by

$$\mathcal{G} \triangleq \left\{ g = \sum_{k=1}^{m} x_k g_k : x \in \mathcal{S}^m \right\}, \tag{4.5}$$

for some choice of distributions g_k, $k = 1, \ldots, m$. The g_k's are often called *dictionary functions* [2]. For example:

- $\mathcal{G} = \mathcal{R}$: $g_k = \delta_{\mu_k}$, where δ_μ denotes a delta-Dirac point mass at μ, and $\mu_1, \ldots, \mu_m \in \mathbb{R}$ is a suitably fine grid of values across the real line.
- $\mathcal{G} = \mathcal{U}_0, \mathcal{G} = \mathcal{SU}_0$: $g_k = \text{Unif}[0, a_k]$, $\text{Unif}[-a_k, 0]$ or $\text{Unif}[-a_k, a_k]$, where $a_1, \ldots, a_m \in \mathbb{R}^+$ is a suitably large and fine grid of values.
- $\mathcal{G} = \mathcal{SN}_0$: $g_k = N(0, \sigma_k^2)$, where $\sigma_1^2, \ldots, \sigma_m^2 \in \mathbb{R}^+$ is a suitably large and fine grid of values.

With these approximations, solving (4.4) reduces to solving an optimization problem of the form (4.1), with $L_{jk} = \int N(z_j; \theta, s_j^2)\, g_k(d\theta)$, the convolution of g_k with a normal density $N(z_j; \theta, s_j^2)$.

A common feature of these examples is that they all use a fine grid to approximate a nonparametric family. The result is that *many of the distributions g_k are similar to one another.* Hence, the matrix L is numerically rank deficient and, in our experience, many of its singular values are near floating-point machine precision. We pay particular attention to this property when designing our optimization methods.

The normal means problem is just one example of a broader class of problems with similar features. The general point is that nonparametric problems can often be accurately solved with finite mixtures, resulting in optimization problems of the form (4.1), typically with moderately large m, large n, and a numerically rank-deficient $n \times m$ matrix L.

4.3 A New Sequential Quadratic Programming Approach

The methods from [90], which are implemented in function `KWDual` from R package `REBayes` [90] provide, to our knowledge, the best current implementation for solving (4.1). These methods are based on reformulating (4.1) as

$$\underset{x \in \mathcal{S}^m}{\text{minimize}} \quad -\frac{1}{n}\sum_{j=1}^{n} \log y_j \quad \text{subject to} \quad Lx = y, \tag{4.6}$$

then solving the dual problem ("K-W dual" in Koenker and Mizera 91),

$$\underset{\nu \in \mathbb{R}^m}{\text{minimize}} \quad -\frac{1}{n}\sum_{j=1}^{n} \log \nu_j \quad \text{subject to} \quad L^T\nu \preceq n\mathbb{1}_m, \quad \nu \succeq 0, \tag{4.7}$$

where $\mathbb{1}_m$ is a vector of ones of length m. [91] reported that solving this dual formulation was generally faster than primal formulations in their assessments. Indeed, we also found this to be the case for IP approaches (see Figure 4.1). For $n \gg m$, however, we believed that the original formulation (4.1) offered more potential for improvement. In the dual formulation (4.7), effort depends on n when computing the gradient and Hessian of the objective, when evaluating the constraints, and when computing the Newton step inside the IP algorithm. By contrast, in the primal formulation (4.1), effort depends on n only in the gradient and Hessian computations; all other evaluations depend on m only.

These considerations motivated the design of our algorithm. It was developed with two key principles in mind: (i) make best possible use of each expensive gradient and Hessian computation in order to minimize the number of gradient and Hessian evaluations; and (ii) reduce the expense of each gradient and Hessian evaluation as much as possible. (We could have avoided Hessian computations by pursuing a first-order optimization method, but we judged that a second-order method would likely be more robust and stable because of the ill-conditioning caused by the numerical rank deficiency of L; we briefly investigate the potential of first-order optimization methods in Section 4.4.3.)

To make effective use of each Hessian computation, we apply sequential quadratic programming

[115] to a reformulation of the primal problem (4.1). SQP attempts to make best use of the expensive Hessian computations by finding, at each iteration, the best reduction in a quadratic approximation to the constrained optimization problem.

To reduce the computational cost of each Hessian evaluation, we use a "rank-revealing" QR decomposition of L to exploit the numerical low-rank of L [62]. The RRQR matrix decomposition, which need be performed only once, reduces subsequent per-iteration computations so that they depend on the numerical rank, r, rather than m. In particular, Hessian computations are reduced from $O(nm^2)$ to $O(nr^2)$.

In addition to these two key principles, two other design decisions were also important to reduce effort. First, we introduced a reformulation that relaxes the simplex constraint to a less restrictive non-negative one, which simplifies computations. Second, based on initial observations that the primal solution is often sparse, we implemented an active set method [115]—one that estimates which entries of the solution are zero (this is the "active set")—to solve for the search direction at each iteration of the SQP algorithm. As we show later, an active set approach effectively exploits the solution's sparsity.

The remaining subsections detail these innovations.

4.3.1 A Reformulation

We transform (4.1) into a simpler problem with less restrictive non-negative constraints using the following definition and proposition.

Definition 4.3.1. *A function $\phi : \mathbb{R}^m_+ \to \mathbb{R}$ is said to be "scale invariant" if for any $c > 0$ there exists a $C \in \mathbb{R}$ such that for any $x \in \mathbb{R}^m_+$ we have $\phi(cx) = \phi(x) - C$.*

Proposition 4.3.1. *Consider the simplex-constrained optimization problem*

$$\underset{x \in \mathcal{S}^m}{\text{minimize}}\ \phi(x), \tag{4.8}$$

where $\phi(x)$ is scale invariant, convex, and nonincreasing with respect to x—that is, $x \succeq y$ (the

componentwise partial ordering) implies $\phi(x) \leq \phi(y)$ *for all* $x \in \mathbb{R}^m_+$. *Let* $x^*(\lambda)$ *denote the solution to a Lagrangian relaxation of* (4.8)*,*

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad \phi_\lambda(x) \triangleq \phi(x) + \lambda \sum_{k=1}^{m} x_k, \quad \text{subject to} \quad x \succeq 0, \tag{4.9}$$

for $\lambda > 0$. *Then* $x^* \triangleq x^*(\lambda) / \sum_{k=1}^m x_k^*(\lambda)$ *is a solution to* (4.8)*.*

By setting ϕ to f, the objective function in (4.1), this proposition implies that (4.1) can be solved by instead solving (4.9) for some λ. Somewhat surprisingly, in this special case setting $\lambda = 1$ yields a solution $x^*(\lambda)$ that is already normalized to sum to 1, so $x^* = x^*(1)$. This result is summarized in the following proposition:

Proposition 4.3.2. *Solving the target optimization problem* (4.1) *is equivalent to solving* (4.9) *with* $\phi = f$ *and* $\lambda = 1$*; that is,*

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad f^\star(x) \triangleq f(x) + \sum_{k=1}^{m} x_k, \quad \text{subject to} \quad x \succeq 0. \tag{4.10}$$

The proofs of the propositions are given in the Supplementary Material.

While we focus on the case of $\phi = f$, these ideas should apply to other objective functions so long as they satisfy Definition 4.3.1; e.g., when f is a composite of "easily differentiable" scale-invariant functions and "thin and tall" linear functions. Many of the algorithmic ideas presented in following sections are applicable to those functions as well. See the Supplementary Material for further discussion.

4.3.2 Sequential Quadratic Programming

We solve the reformulated optimization problem (4.10) using an SQP algorithm with backtracking line search [115]. In brief, SQP is an iterative algorithm that, at the t-th iteration, computes a second-order approximation of the objective at the feasible point $x^{(t)}$, then determines a search direction $p^{(t)}$ based on the second-order approximation. At iteration t, the search direction $p^{(t)}$ is

the solution to the following non-negatively-constrained quadratic program:

$$p^{(t)} = \underset{p \in \mathbb{R}^m}{\operatorname{argmin}} \tfrac{1}{2} p^T H_t p + p^T g_t \quad \text{subject to} \quad x^{(t)} + p \succeq 0, \tag{4.11}$$

where $g_t = \nabla f^\star(x^{(t)})$ and $H_t = \nabla^2 f^\star(x^{(t)})$ are the gradient and Hessian of $f^\star(x)$ at $x^{(t)}$. Henceforth, this is called the "quadratic subproblem." Computation of the gradient and Hessian is considered in Section 4.3.3, and solving the quadratic subproblem is discussed in Section 4.3.4.

After identifying the search direction, $p^{(t)}$, the SQP method performs a backtracking line search to determine a sufficient descent step $x^{(t+1)} = x^{(t)} + \alpha_t p^{(t)}$, for $\alpha_t \in (0, 1]$. In contrast to other projection-based methods such as the projected Newton method [86], $x^{(t+1)}$ is guaranteed to be (primal) feasible for all choices of $\alpha_t \in (0, 1]$ provided that $x^{(t)}$ is feasible. This is due to the linearity of the inequality constraints. As discussed in [115], the line search will accept unitary steps ($\alpha_t = 1$) close to the solution and the iterates will achieve quadratic convergence provided the reduced Hessian is positive definite for all co-ordinates outside the optimal active set (the indices that are zero at the solution x^*). A similar result can be found in [168].

4.3.3 Gradient and Hessian Evaluations

We now discuss computation of the gradient and Hessian and ways to reduce the burden of computing them.

Lemma 4.3.1. *For any $x \in \mathbb{R}^m_+$, the gradient and Hessian of the objective function in* (4.10) *are given by*

$$g = \nabla f^\star(x) = -\tfrac{1}{n} L^T d + \mathbb{1}_m, \quad H = \nabla^2 f^\star(x) = \tfrac{1}{n} L^T \operatorname{diag}(d)^2 L, \tag{4.12}$$

where $\mathbb{1}_m$ is a vector of ones of length m, and $d = (d_1, \ldots, d_n)^T$ is a column vector with entries $d_j = 1/(Lx)_j$. Further, for any $x \in \mathbb{R}^m_+$ the following identities hold:

$$x^T g = 0, \quad x^T H x = 1, \quad Hx + g = \mathbb{1}_m. \tag{4.13}$$

Assuming L is not sparse, computing g (and d) requires $O(nm)$ multiplications, and computing H requires $O(nm^2)$ multiplications. The result (4.13) is easily derived by substitution from (4.12).

In practice, we find that L is often numerically rank deficient, with (numerical) rank $r < m$. We can exploit this property to reduce computational effort by approximating L with a low-rank matrix. We use either the RRQR decomposition [62] or a truncated singular value decomposition (tSVD) to compute an accurate, low-rank approximation to L. Specifically, we use the `pqrfact` and `psvdfact` functions from the `LowRankApprox` Julia package [74] which implement randomized algorithms based on [68].

The rank-r QR approximation of L is $\tilde{L} = QRP^T$, with $Q \in \mathbb{R}^{n\times r}$, $R \in \mathbb{R}^{r\times m}$ and $P \in \mathbb{R}^{m\times m}$, the permutation matrix. Therefore, a rank-r QR decomposition yields an approximate gradient and Hessian:

$$\tilde{g} = -\tfrac{1}{n} PR^T Q^T d + \mathbb{1}_m, \quad \tilde{H} = \tfrac{1}{n} PR^T Q^T \mathrm{diag}(d)^2 QRP^T, \tag{4.14}$$

where $\tilde{d}$ is a vector of length n with entries $\tilde{d}_j = 1/(QRP^T x)_j$. Corresponding expressions for the tSVD are straightforward to obtain, and are therefore omitted.

Once we have obtained a truncated QR (or SVD) approximation to L, the key to reducing the expense of the gradient and Hessian computations is to avoid directly reconstructing $\tilde{L}$. For example, computing $\tilde{g}$ is implemented as $\tilde{g} = -(((d^T Q)R)P^T)^T/n + \mathbb{1}_m$. In this way, all matrix multiplications are a matrix times a vector. The dominant cost in computing $\tilde{H}$ is the product $Q^T \mathrm{diag}(d)^2 Q$, which requires $O(nr^2)$ multiplications. Overall, computation is reduced by roughly a factor of $(r/m)^2$ per iteration compared with (4.12). To enjoy this benefit, we pay the one-time cost of factorizing L, which, in the regime $n \gg m$, is $O(nmr)$ [62].

4.3.4 Solving the Quadratic Subproblem

To find the solution to the quadratic subproblem (4.11), we set $p^* = y^* - x^{(t)}$ in which y^* is the solution to

$$\underset{y\in\mathbb{R}^m}{\text{minimize}}\ \tfrac{1}{2} y^T H_t y + y^T a_t, \quad \text{subject to} \quad y \succeq 0 \tag{4.15}$$

with $a_t = -H_t x^{(t)} + g_t = 2g_t - \mathbb{1}_m$. This problem comes from substituting $y = x^{(t)} + p$ into (4.11). Solving (4.15) is easier than (4.11) due to the simpler form of the inequality constraints.

To solve (4.15), we implement an active set method following Nocedal and Wright [115, §16.5]. The active set procedure begins at a feasible point $y^{(0)}$ and an initial estimate of the active set, $\mathcal{W}^{(0)}$ (the "working set"), and stops when the iterates $y^{(0)}, y^{(1)}, y^{(2)}, \ldots$ converge to a fixed point of the inequality-constrained quadratic subproblem (4.15). The initial working set $\mathcal{W}^{(0)}$ and estimate $y^{(0)}$ can be set to predetermined values, or they can be set according to the current SQP iterate, $x^{(t)}$ (this is often called "warm-starting"). We initialized the active set solver to $x^{(0)} = \{\frac{1}{m}, \ldots, \frac{1}{m}\}$ in the first iteration of SQP, and used a warm start in subsequent iterations.

The active set method is an iterative method in which the lth iteration involves solving an equality-constrained problem,

$$\underset{q \in \mathbb{R}^m}{\text{minimize}} \ \tfrac{1}{2} q^T H_t q + q^T b_l \quad \text{subject to} \quad q_i = 0, \ \forall i \in \mathcal{W}^{(l)}, \tag{4.16}$$

where $b_l = H_t y^{(l)} + a_t$. The solution to (4.16) at the lth iteration, $q^{(l)}$, yields a search direction for the next iterate, $y^{(l+1)}$. Computing the solution to (4.16) reduces to solving a system of linear equations, with one equation for each co-ordinate outside the working set. Therefore, if the number of inactive co-ordinates remains much smaller than m, we expect the complexity of the active set step to be much smaller than $O(m^3)$.

Additional details of the active set implementation, including the updates to the working set in the presence of blocking constraints, are given in the Supplementary Material.

4.3.5 The **MIX-SQP** *Algorithm*

Putting these components together results in Algorithm 6, which we call MIX-SQP. We give the algorithm for the case when RRQR is used to approximate L; variants of MIX-SQP using the truncated SVD, or with no approximation to L, are similar. The most complicated part to implement is the active set method; the details of this step are given separately in Algorithm 7 in the Supplementary

Algorithm 6 MIX-SQP with RRQR approximation of L.

Input: likelihood matrix, $L \in \mathbb{R}^{n \times m}$; initial stimate, $x^{(0)} \in \mathcal{S}_m$;
stringency of sufficient decrease condition, $0 < \xi < 1$ (default is 0.01);
step size reduction in backtracking line search, $0 < \rho < 1$ (default is 0.5);
SQP convergence tolerance, $\epsilon_{\text{dual}} \geq 0$ (default is 10^{-8});
convergence tolerance for active set step, $\epsilon_{\text{active-set}} \geq 0$ (default is 10^{-10}).

Output: $x^{(t)} \in \mathbb{R}^m$, an estimate of the solution to (4.1).

Compute RRQR factorization $\{Q, R, P\}$ of L;

for t = 0, 1, 2, ... **do**
  Compute approximate gradient $\tilde{g}_t$ and Hessian $\tilde{H}_t$ at $x^{(t)}$; (see (4.14))
  $\mathcal{W} \leftarrow \{1, \ldots, m\} \setminus \text{supp}(x^{(t)})$; (current estimate of working set)
  $y \leftarrow$ MIX-ACTIVE-SET$(\tilde{g}_t, \tilde{H}_t, \mathcal{W}, \epsilon_{\text{active-set}})$; (see Algorithm 7)
  $p^{(t)} \leftarrow y - x^{(t)}$;
  if $\min_k(\tilde{g}_t)_k \geq -\epsilon_{\text{dual}}$ **then**
    stop; (maximum dual residual of KKT conditions is less than ϵ_{dual})
  end if
  $\alpha_t \leftarrow 1$; (backtracking line search)
  repeat
    $\alpha_t \leftarrow \rho\alpha_t$;
  until $\tilde{f}(x^{(t)} + \alpha_t p^{(t)}) > \tilde{f}(x^{(t)}) + \xi\alpha_t \tilde{g}_t^T p^{(t)}$
  $x^{(t+1)} \leftarrow x^{(t)} + \alpha_t p^{(t)}$;
end for

Material.

4.3.6 Practical Implementation Details

A useful property of problem (4.1) is that the gradient (4.12) is unaffected by the "scale" of the problem; for example, if we multiply all entries of L by 100, the gradient remains the same. This property has several practical benefits; for example, the "dual residual," used to assess convergence of the iterates, is invariant to the scale of L.

When we replace L with an approximation, for example $\tilde{L} = QRP^T$, we are effectively solving an approximation to (4.10),

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad \tilde{f}(x) \triangleq -\frac{1}{n}\sum_{j=1}^{n} \log\left(\sum_{k=1}^{m} \tilde{L}_{jk} x_k\right) + \sum_{k=1}^{m} x_k \quad \text{subject to} \quad x \succeq 0. \tag{4.17}$$

When an approximated likelihood matrix $\tilde{L}$ is used, some entries $\tilde{L}_{jk}$ may have negative values, and thus the terms inside the logarithms, $\sum_{k=1}^{m} \tilde{L}_{jk} x_k$, can be slightly below zero at some feasible points. This can occur either due to the randomized nature of the matrix decomposition algorithms we used, or due to the finite precision of the low-rank approximations. This is a critical point to attend to since the logarithm in the objective does not accept negative values, for example when the objective of (4.17) is evaluated during line search. In principle, this is not a problem so long as the initial point $x^{(0)}$ satisfies $\sum_{k=1}^{m} \tilde{L}_{jk} x_k^{(0)} > 0$ for all j. Indeed, one can refine the problem statement as

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad \tilde{f}(x) \quad \text{subject to} \quad x \succeq 0,\ \tilde{L}x \succeq 0, \tag{4.18}$$

and start with a feasible $x^{(0)}$. In practice, we implemented a simple workaround: we added a small positive constant (typically somewhere between 10^{-8} and 10^{-6}) to all the terms inside the logarithms so as to ensure that they were strictly bounded away from zero for all $x \in \mathcal{S}^m$.

4.4 Numerical Experiments

We conducted numerical experiments to compare different methods for solving problems of the form (4.1). We considered problems of this form that arise from nonparametric EB, with $\mathcal{G} = \mathcal{SN}_0$ (Section 4.2). Our comparisons involved simulating data sets with varying numbers of data points (n) and grid sizes (m). We also evaluated the methods on a large-scale genetic data set.

4.4.1 Real and Synthetic Data Sets

For each synthetic data set, we generated $z_1, \ldots, z_n$ independently as

$$z_j \,|\, \theta_j \sim N(\theta_j, 1),$$

where the means θ_j were *i.i.d.* random draws from g, a heavy-tailed symmetric distribution about zero,

$$g = 0.5N(0,1) + 0.2t_4 + 0.3t_6.$$

Here, t_ν denotes the density of Student's t distribution with ν degrees of freedom.

In addition, we used data generated by the GIANT consortium ("Genetic Investigation of ANthropometric Traits") for investigating the genetic basis of human height [167]. We used the additive effects on height estimated for $n = 2{,}126{,}678$ genetic variants (single nucleotide polymorphisms, or SNPs) across the genome. Height is a well-studied example of trait with a complex genetic basis, so the distribution of additive effects is expected to be representative of genetic associations for many other complex traits and diseases [103]. The data consist of the estimated effect sizes, z_j, and their corresponding standard errors, s_j. For illustration, we treat the n data points as independent, though in practice there are local correlations between SNPs. See the Supplementary Material for more details on steps taken to download and prepare the GIANT data.

We modeled all data sets using the "adaptive shrinkage" nonparametric EB model from Stephens [143] (see Sec. 4.2). The R package `ashr`, available on CRAN [145], implements various versions of adaptive shrinkage; for our experiments, we re-coded the simplest version of adaptive shrinkage in Julia. This assumes a normal likelihood and a prior that is a finite mixture of zero-centered normals ($\mathcal{G} = \mathcal{SN}_0$ in Sec. 4.2). This leads to matrix entries $L_{jk} = N(z_j; 0, \sigma_k^2 + s_j^2)$, the normal density at z_j with zero mean and variance $\sigma_k^2 + s_j^2$, where σ_k^2 is the variance of the kth mixture component. For the simulated data, we set $s_j = 1$; for the GIANT data, we set the s_j's to the standard errors provided. The adaptive shrinkage method also requires a grid of variances, $\sigma_1^2, \ldots, \sigma_m^2$. We ran MIX-SQP on matrices L with a range of grid sizes, m, and for each grid size we used the method from [143] to select the grid values. To avoid numerical overflow or underflow, each row of L was computed up to a constant of proportionality such that the largest entry in each row was always 1. (Recall the maximum likelihood estimate is invariant to scaling the rows of L.)

4.4.2 Approaches Considered

Most optimization methods tested in our experiments are combinations of the following four elements:

1. **Problem formulation:** The method either solves the dual (4.7), simplex-constrained (4.1), or non-negatively-constrained formulation (4.10). This choice is indicated by `D`, `S`, or `NN`, respectively.

2. **Solver:** The optimization problem is solved using an SQP or IP solver. This choice is denoted by `SQP` and `IP`, respectively.

3. **QP solver:** For SQP approaches only, we considered two ways to solve the quadratic subproblem (4.11): an active set method (see Section 4.3.4) or an off-the-shelf QP solver (the commercial IP solver `MOSEK`). We indicate this choice with `A` or `IP`, respectively. When the SQP method is not used, we indicate this choice with `NA`, for "not applicable."

4. **Gradient and Hessian computation:** The objective and partial derivatives either are computed exactly (within floating-point precision) by using the full matrix L, or approximated using a truncated SVD or RRQR decomposition of L (Section 4.3.3). We denote this choice by `F` (for "full matrix"), `SVD` or `QR`.

An optimization method is therefore fully specified by

`[formulation]-[solver]-[QP solver]-[gradient/Hessian computation].`

For example, the MIX-SQP method with a RRQR approximation L (Algorithm 6) is written as `NN-SQP-A-QR`. We also assessed the performance of two methods that do not use second-order information: projected gradient descent [15, 135] and EM (see Supplementary Material 4.6.1).

All numerical comparisons were run on machines with an Intel Xeon E5-2680v4 ("Broadwell") processor. Other than the projected gradient method, all methods were run from Julia version 0.6.2 [13] linked to the OpenBLAS optimized numerical libraries that were distributed with Julia. (The code has also been updated for Julia 1.1.) The `KWDual` function in R package `REBayes` [90] was

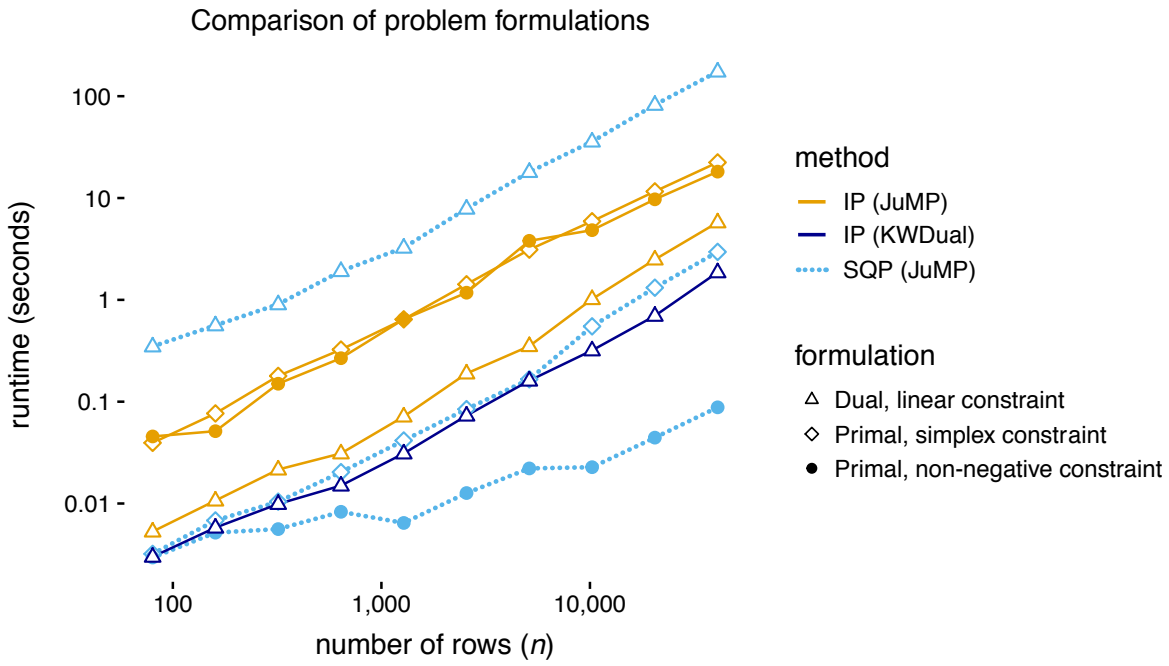


Figure 4.1: Runtimes for different formulations of the maximum likelihood estimation problem: dual (4.7), simplex-constrained (4.1) and non-negatively-constrained (4.10). For each problem formulation, we applied an IP or SQP algorithm. As a baseline, we compared against the `KWDual` function from the `REBayes` package which solves the dual formulation using `MOSEK`. Results are from data sets with $m = 40$ and n varying from 80 to 40,960. Runtimes are averages over 10 independent simulations.

called in R 3.4.1 and was interfaced to Julia using the `RCall` Julia package. The KWDual function used version 8.0 of the MOSEK optimization library. The projected gradient method was run in MATLAB 9.5.0 (2018b) with optimized Intel MKL numerical libraries. Julia source code and scripts implementing the methods compared in our experiments, including Jupyter notebooks illustrating how the methods are used, are available at `https://github.com/stephenslab/mixsqp-paper`. The MIX-SQP method is also implemented as an R package, `mixsqp`, which is available on CRAN and GitHub (`https://github.com/stephenslab/mixsqp`).

4.4.3 Numerical Results

Comparison of problem formulations

First, we investigated the benefits of the three problem formulations: the dual form (4.7), the simplex-constrained form (4.1) and the non-negatively-constrained form (4.10). We implemented IP and SQP methods for each of these problem formulations in the `JuMP` modeling environment [39]. We applied these methods to $n \times m$ simulated data sets L, with $m = 40$ and n ranging from 80 to 40,960. For all SQP solvers, an IP method was used to solve the quadratic subproblems. In summary, we compared solvers x`-IP-NA-F` and x`-SQP-IP-F`, substituting `D`, `S` or `NN` for x. In all cases, the commercial solver `MOSEK` was used to implement the IP method. To provide a benchmark for comparison, we also ran the `KWDual` method in R, which calls `MOSEK`.

The results of this experiment are summarized in Figure 4.1. All runtimes scaled approximately linearly in n (the slope is near 1 on the log-log scale). Of the methods compared, SQP applied to the non-negatively-constrained formulation, `NN-SQP-IP-F`, consistently provided the fastest solution.

SQP for the non-negatively-constrained formulation was substantially faster than SQP for the simplex-constrained formulation. The former typically required fewer outer iterations, but this does not completely explain the difference in performance—it is possible that the simplex-constrained formulation could be improved with more careful implementation using the `JuMP` interface.

Of the IP approaches, the fastest was the implementation using the dual formulation. This is consistent with the results of [91]. By contrast, the SQP approach appears to be poorly suited to the dual formulation.

Based on these results, we focused on the non-negatively-constrained formulation for SQP in subsequent comparisons.

Examining the benefits of approximating L

Next, we investigated the benefits of exploiting the low-rank structure of L (see Section 4.3.3). We compared the runtime of the SQP method with and without low-rank approximations, RRQR and

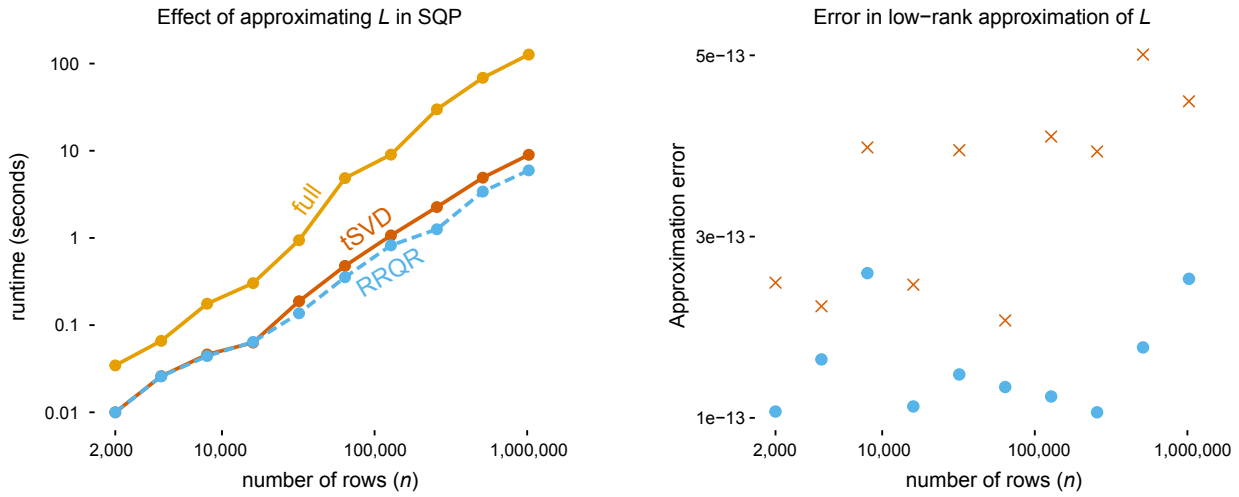


Figure 4.2: Comparison of SQP methods with and without low-rank approximations to the $n \times m$ matrix, L. *(Left)* Runtime of the SQP solver using the full matrix ("full") and using low-rank approximations based on RRQR and tSVD factorizations. Results are from data sets with $m = 100$ and varying n. *(Right)* Reconstruction error in RRQR (blue circles) and tSVD (red crosses); error is computed as $\|L - \tilde{L}\|_F$. All runtimes and errors are averages over 10 simulations.

tSVD; that is, we compared solvers `NN-SQP-A-`x, with x being one of `F`, `QR`, or `SVD`. We applied the three SQP variants to L matrices with varying numbers of rows. We used functions `pqrfact` and `psvdfact` from the `LowRankApprox` Julia package [74] to compute the RRQR and tSVD factorizations. For both factorizations, we set the relative tolerance to 10^{-10}.

The left-hand panel in Figure 4.2 shows that the SQP method with an RRQR and tSVD approximation of L was consistently faster than running SQP with the full matrix, and by a large margin; e.g., at $n = 10^6$, the runtime was reduced by a factor of over 10. At the chosen tolerance level, these low-rank approximations accurately reconstructed the true L (Figure 4.2, right-hand panel).

For the largest n, SQP with RRQR was slightly faster than SQP with the tSVD. We attribute this mainly to the faster computation of the RRQR factorization. We found that the SQP method took nearly the same solution path regardless of the low-rank approximation method used (results not shown).

The demonstrated benefits in using low-rank approximations are explained by the fact that r, the effective rank of L, was small relative to m in our simulations. To check that this was not a particular

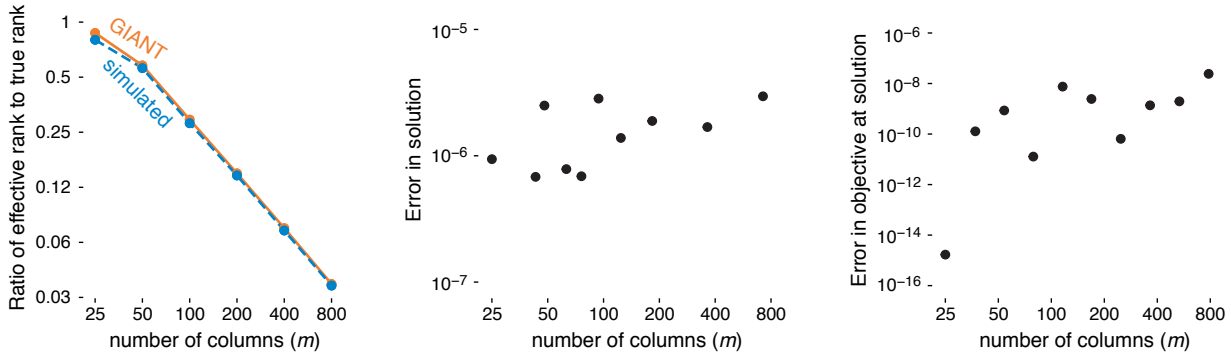


Figure 4.3: Assessment of numeric rank of L and its effect on solution accuracy. *(Left)* The ratio of the effective rank r (the rank estimated by `pqrfact`, with relative tolerance of 10^{-10}) to m, the number of columns. The ratios for the simulated data sets are averages from 10 simulations. *Middle panel:* ℓ_1-norm of the differences in the solutions from the `NN-SQP-A-F` and `NN-SQP-A-QR` solvers applied to the GIANT data set. *(Right)* Absolute difference in the objective values at these solutions. For all data sets used in these experiments, $n = 2{,}126{,}678$.

feature of our simulations, we applied the same SQP method with RRQR (`NN-SQP-A-QR`) to the GIANT data set. The ratio r/m in the simulated and genetic data sets is nearly the same (Figure 4.3, left-hand panel).

We also assessed the impact of using low-rank approximations on the quality of the solutions. For these comparisons, we used the RRQR decomposition and the GIANT data set. In all settings of m tested, the error in the solutions was very small; the ℓ_1-norm in the difference between the solutions with exact and approximate L was close to 10^{-6} (Figure 4.3, middle panel), and the difference in the objectives was always less than 10^{-8} in magnitude (Figure 4.3, right-hand panel).

To further understand how the RRQR approximation of L affects solution accuracy, we re-ran the SQP solver using QR approximations with different ranks, rather than allow the "rank revealing" QR algorithm to adapt the rank to the data. Figure 4.4 shows that the quality of the approximate solution varies greatly depending on the rank of the QR decomposition, and that the approximate solution gets very close to the unapproximated solution as the rank is increased (presumably as it approaches the "true" rank of L). These results illustrate the importance of allowing the RRQR algorithm to adapt the low-rank factorization to the data.

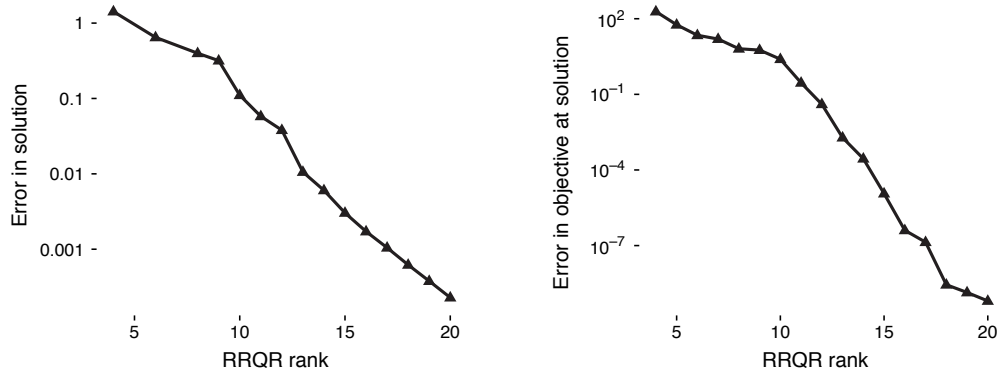


Figure 4.4: Effect of QR rank (r) on accuracy of SQP solution. *Left panel:* ℓ_1-norm of the difference in the solutions from the `NN-SQP-A-F` and `NN-SQP-A-QR` solvers applied to the same matrix L. *(Right)* Absolute difference in the objective values at these solutions. All results are averaged over 10 simulated data sets with $n = 100{,}000$, $m = 200$ and r ranging from 4 to 20.

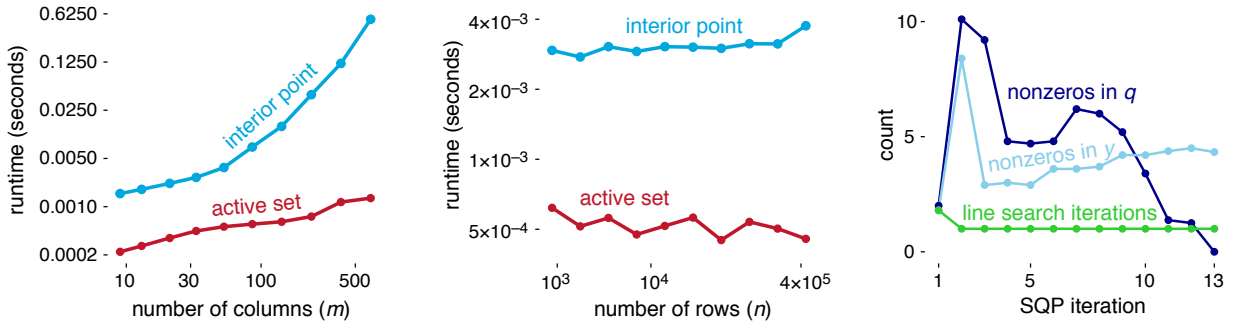


Figure 4.5: Comparison of active set and interior point methods for solving the SQP quadratic subproblems. *(Left)* Runtimes of the active set and IP (`MOSEK`) methods as m varies, with $n = 10^5$. Runtimes are averaged over all subproblems solved, and over 10 simulations. *(Center)* Runtimes of the IP and active set methods as n varies, with $m = 40$. *(Right)* Number of backtracking line searches, and the number of nonzero entries in q and y at each iteration of SQP, averaged over all SQP iterations and all 10 simulations. (Refer to (4.15) and (4.16) for interpreting q and y.)

Comparison of active set and IP solutions to quadratic subproblem

In this set of experiments, we compared different approaches to solving the quadratic subproblems inside the SQP algorithm: an active set method (Section 4.3.4) and an off-the-shelf IP method (`MOSEK`); specifically, we compared `NN-SQP-A-F` against `NN-SQP-IP-F`. To assess effort, we recorded only the time spent in solving the quadratic subproblems.

The left and middle plots in Figure 4.5 show that the active set method was consistently faster than the IP method by a factor of roughly 5 or more, with the greatest speedups achieved when m and n were large. For example, when $n = 10^5$ and $m \approx 500$ in the left-hand plot, the active set solver was over 100 times faster than the IP method on average. The left-hand plot in Figure 4.5 shows that the complexity of the active set solution to the quadratic subproblem grows linearly in m, whereas the complexity of the IP solution grows quadratically. By contrast, the average time required to solve the quadratic subproblems does not depend on n (see Figure 4.5, middle panel), which could be explained by n having little to no effect on the number of degrees of freedom (sparsity) of the solution x^*.

We hypothesize that the active set method is faster because the quadratic subproblem iterates and final solution are sparse (see the right-hand plot in Figure 4.5 for an illustration). Recall that the reformulated problem (4.10) and the quadratic subproblem (4.11) have a non-negative constraint, which promotes sparsity.

Based on these results, we infer that the active set method effectively exploits the sparsity of the solution to the quadratic subproblem. We further note that poor conditioning of the Hessian may favor the active set method because it tends to search over sparse solutions where the reduced Hessian is better behaved.

In addition to the runtime improvements of the active set method, another benefit is that it is able to use a good initial estimate when available ("warm starting"), whereas this is difficult to achieve with IP methods. Also, our active set implementation has the advantage that it does not rely on a commercial solver. These qualitative benefits may in fact be more important than performance improvements considering that the fraction of effort spent on solving the quadratic subproblems (either using the active set or IP methods) is relatively small when $n \gg m$.

Comparison of **mix-SQP** and KWDual

Based on the numerical results above, we concluded that when n and m are large, and n is larger than m, the fastest approach is `NN-SQP-A-QR`. We compared this approach, which we named MIX-SQP,

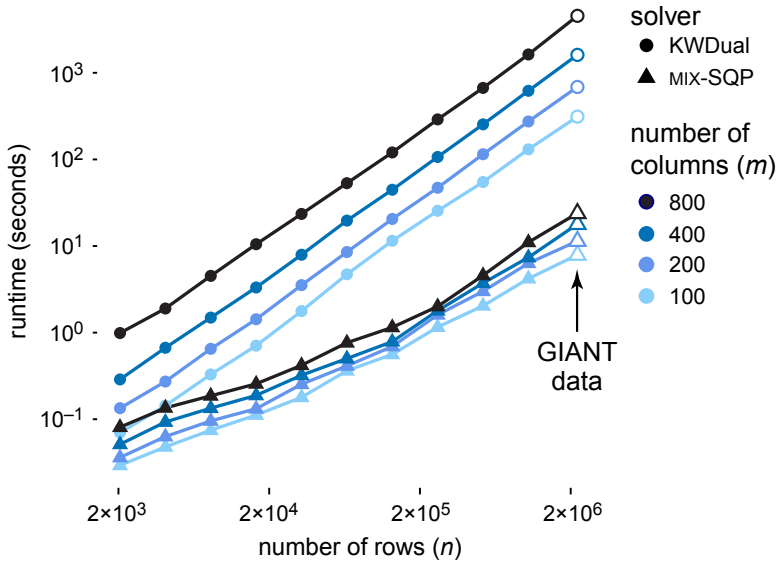


Figure 4.6: Runtimes of MIX-SQP and `KWDual` (which uses `MOSEK`) applied to simulated data sets with varying n and m, and to the GIANT data set (n = 2,126,678). All runtimes on simulated data sets were taken as averages over 10 separate simulations. Each timing on the GIANT data set is averaged over 10 independent runs with the same L matrix.

against the `KWDual` function from the R package `REBayes` [90], which is a state-of-the-art solver that interfaces to the commercial software `MOSEK` (this is `D-IP-NA-F`). For fair comparison, all timings of `KWDual` were recorded in R so that communication overhead in passing variables between Julia and R was not factored into the runtimes.

Although R often does not match the performance of Julia, an interactive programming language that can achieve computational performance comparable to C [13], `KWDual` is fast because most of the computations are performed by `MOSEK`, an industry-grade solver made available as an architecture-optimized dynamic library. Therefore, it is significant that our Julia implementation consistently outperformed `KWDual` in both the simulated and genetic data sets; see Figure 4.6. For the largest data sets (e.g., $n \approx 10^6, m = 800$), MIX-SQP was over 100 times faster than `KWDual`. Additionally, `KWDual` runtimes increased more rapidly with m because `KWDual` did not benefit from the reduced-rank matrix operations.

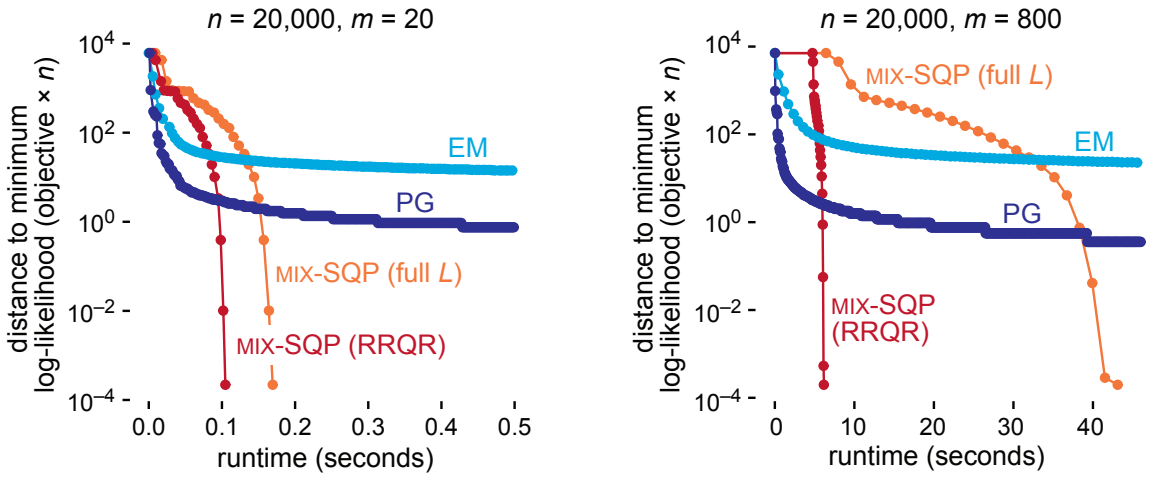


Figure 4.7: Progress over time for the EM, projected gradient and MIX-SQP methods on two simulated data sets with $n = 20{,}000$ rows and $m = 20$ or $m = 800$ columns. The vertical axis shows the difference between the value of the log-likelihood, $f(x) \times n$, at the current iterate x, and the value of the log-likelihood at the best solution. Each dot corresponds to a single iteration of the algorithm's main loop.

Assessing the potential for first-order optimization

Our primary focus has been the development of fast methods for solving (4.1), particularly when n is large. For this reason, we developed a method, MIX-SQP, that makes best use of the second-order information to improve convergence. However, it is natural to ask whether MIX-SQP is an efficient solution when m is large; the worst-case complexity is $O(m^3)$ since the active set step requires the solution to a system of linear equations as large as $m \times m$. (In practice, the complexity is often less that this worst case because many of the co-ordinates are zero along the solution path.) Here, we compare MIX-SQP against two alternatives that avoid the expense of solving an $m \times m$ linear system: a simple projected gradient algorithm [135], in which iterates are projected onto the simplex using a fast projection algorithm [38]; and EM [33], which can be viewed as a gradient-descent method [172].

The projected gradient method was implemented using Mark Schmidt's MATLAB code.[1] In brief, the projected gradient method is a basic steepest descent algorithm with backtracking line

1. `minConF_SPG.m` was retrieved from `https://www.cs.ubc.ca/spider/schmidtm/Software/minConf.html`.

search, in which the steepest descent direction $-\nabla f(x)$ at iterate x is replaced by $\mathcal{P}_{\mathcal{S}}(x-\nabla f(x))-x$, where $\mathcal{P}_{\mathcal{S}}$ denotes the projection onto the feasible set $\mathcal{S}^m$. As expected, we found that the gradient descent steps were often poorly scaled, resulting in small step sizes. We kept the default setting of 1 for the initial step size in the backtracking line search. (The spectral gradient method, implemented in the same MATLAB code, is supposed to improve the poor scaling of the steepest descent directions, but we found that it was unstable for the problems considered here.) The EM algorithm, which is very simple (see Supplementary Material 4.6.1), was implemented in Julia.

To illustrate the convergence behaviour of the first-order methods, we ran the approaches on two simulated data sets and examined the improvement in the solution over time. Our results on a smaller (20,000 $\times$ 20) and a larger (20,000 $\times$ 800) data set are shown in Figure 4.7. Both first-order methods show a similar convergence pattern: initially, they progress rapidly toward the solution, but this convergence slows considerably as they approach the solution; for example, even after running EM and projected gradient for 1,000 iterations on the 20,000 $\times$ 800 data set, the solution remained at least 0.35 log-likelihood units away from the solution obtained by MIX-SQP. Among the two first-order methods, the projected gradient method is clearly the better option. Relative to the first-order methods, each iteration of MIX-SQP is very costly—initial iterations are especially slow because the iterates contain few zeros at that stage—but MIX-SQP is able to maintain its rapid progress as the iterates approach the solution. The benefit of the low-rank (RRQR) approximation in reducing effort is particularly evident in the larger data set.

Although this small experiment is not intended to be a systematic comparison of first-order vs. second-order approaches, it does provide two useful insights. One, second-order information is crucial for obtaining good search directions near the solution. Two, gradient-descent methods are able to rapidly identify good approximate solutions. These two points suggest that more efficient solutions for large data sets, particularly data sets with large m, could be achieved using a combination of first-order and second-order approaches, or using quasi-Newton methods.

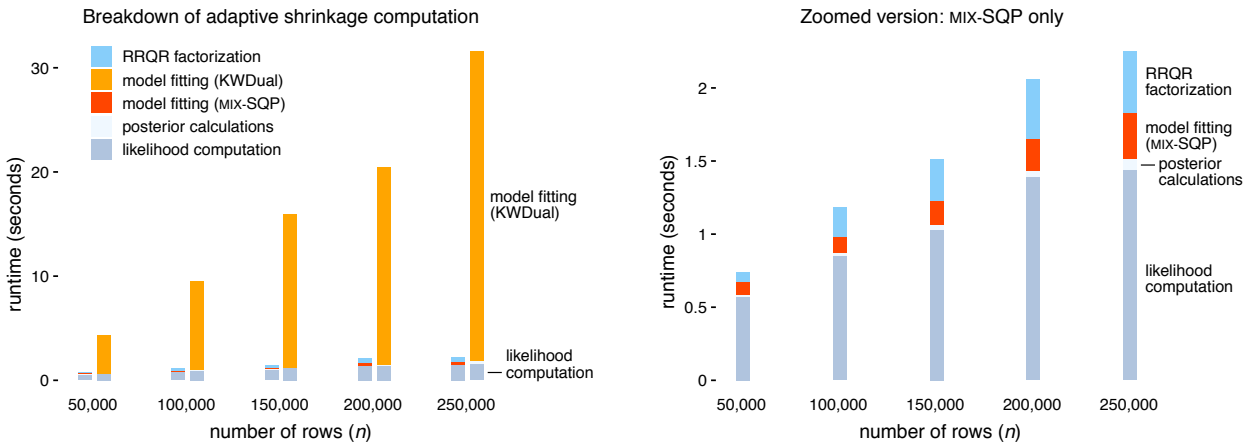


Figure 4.8: Breakdown of computations for the "adaptive shrinkage" EB method [143], in which the model-fitting step (*i.e.*, maximum likelihood estimation of the mixture proportions) was implemented with either MIX-SQP or `KWDual` (`MOSEK`). The adaptive shrinkage method was applied to simulated data sets with $m = 100$ and varying n. All runtimes were averaged over 10 independent simulations. The right-hand panel is a zoomed-in version of the MIX-SQP results shown in the left-hand plot.

Profiling adaptive shrinkage computations

An initial motivation for this work was our interest in applying a nonparametric EB method, "adaptive shrinkage," to very large data sets. This EB method involves three steps: (1) likelihood computation; (2) maximum likelihood estimation of the mixture proportions; and (3) posterior computation. When we began this work, the second step, solved using `MOSEK`, was the computational bottleneck of our R implementation [145]; see Figure 4.8, which reproduces the adaptive shrinkage computations in Julia (aside from `KWDual`). (To verify that this bottleneck was not greatly impacted by the overhead of calling `MOSEK` from R inside function `KWDual`, we also recorded runtime estimates outputted directly by `MOSEK`, stored in output `MSK_DINF_OPTIMIZER_TIME`. We found that the overhead was at most 1.5 s, a small fraction of the total model-fitting time under any setting shown in Figure 4.8. Note all timings of `KWDual` called from Julia were recorded in R, not Julia.)

When we replaced `KWDual` with MIX-SQP, the model-fitting step no longer dominated the computation time (Figure 4.8). This result is remarkable considering that the likelihood calculations for the scale mixtures of Gaussians involve relatively simple probability density computations.

4.5 Conclusions and Potential Extensions

We have proposed a combination of optimization and linear algebra techniques to accelerate maximum likelihood estimation of mixture proportions. The benefits of our methods are particularly evident at settings in which the number of mixture components, m, is moderate (up to several hundred) and the number observations, n, is large. In such settings, computing the Hessian is expensive—$O(nm^2)$ effort—much more so than Cholesky factorization of the Hessian, which is $O(m^3)$. Based on this insight, we developed a sequential quadratic programming approach that makes best use of the (expensive) gradient and Hessian information, and minimizes the number of times it is calculated. We also used linear algebra techniques, specifically the RRQR factorization, to reduce the computational burden of gradient and Hessian evaluations by exploiting the fact that the matrix L often has a (numerically) low rank. These linear algebra improvements were possible by developing a customized SQP solver, in contrast to the use of a commercial (black-box) optimizer such as MOSEK. Our SQP method also benefits from the use of an active set algorithm to solve the quadratic subproblem, which can take advantage of sparsity in the solution vector. The overall result is that for problems with $n > 10^5$, MIX-SQP can achieve a 100-fold speedup over `KWDual`, which applies the commercial `MOSEK` interior point solver to a dual formulation of the problem.

To further reduce the computational effort of optimization in nonparametric EB methods such as [91] and [143], *quasi-Newton methods* may be fruitful [115]. Quasi-Newton methods, including the most popular version, BFGS, approximate H by means of a secant update that employs derivative information without ever computing the Hessian. While such methods may take many more iterations compared with exact Hessian methods, their iterations are m times cheaper—consider that evaluating the gradient (4.14) is roughly m times cheaper than the Hessian when $n \gg m$—and, under mild conditions, quasi-Newton methods exhibit the fast superlinear convergence of Newton methods when sufficiently close to the solution [115].

Since n is the dominant component of the computational complexity in the problem settings we explored—the gradient and Hessian calculations scale linearly with n—another promising direction is the use of *stochastic approximation* or *online learning* methods, which can often achieve good

solutions using approximate gradient (and Hessian) calculations that do not scale with n [21, 127]. Among first-order methods, *stochastic gradient descent* may allow us to avoid linear per-iteration cost in n. In the Newton setting, one could explore stochastic quasi-Newton [26] or LiSSA [1] methods.

As we briefly mentioned in the results above, an appealing feature of SQP approaches is that they can easily be warm started. This is much more difficult for interior point methods [121]. "Warm starting" refers to sequential iterates of a problem becoming sufficiently similar that information about the subproblems that is normally difficult to compute from scratch ("cold") can be reused as an initial estimate of the solution. The same idea also applies to solving the quadratic subproblems. Since, under general assumptions, the active set settles to its optimal selection before convergence, this suggests that the optimal working set $\mathcal{W}$ for subproblem $\mathcal{P}$ will often provide a good initial guess for the optimal working set $\mathcal{W}^\star$ for similar subproblem $\mathcal{P}^\star$.

4.6 Supplementary Materials

4.6.1 EM for Maximum-likelihood Estimation of Mixture Proportions

Here we derive the EM algorithm for solving (4.1). The objective $f(x)$ can be recovered as the log-likelihood (divided by n) for $z_1, \ldots, z_n$ drawn *i.i.d.* from a finite mixture, in which the x_k's specify the mixture weights:

$$p(z_j \,|\, x) \sim \sum_{k=1}^{m} x_k g_k(z_k), \quad \text{for } j = 1, \ldots, n.$$

The mixture model is equivalently formulated as

$$\begin{aligned} p(\gamma_j = k \,|\, x) &= x_k \\ p(z_j \,|\, \gamma_j = k) &= g_k(z_j), \quad \text{for } j = 1, \ldots, n, \end{aligned}$$

in which we have introduced latent indicator variables $\gamma_1, \ldots, \gamma_n$, with $\gamma_j \in \{1, \ldots, m\}$.

Under this augmented model, the expected complete log-likelihood is

$$\begin{aligned} E[\log p(z \,|\, x, \gamma)] &= \sum_{j=1}^{n} E\big[\log \big\{p(z_j \,|\, \gamma_j)\, p(\gamma_j \,|\, x)\big\}\big] \\ &= \sum_{j=1}^{n} \sum_{k=1}^{K} \phi_{jk} \log(L_{jk} x_k), \end{aligned}$$

where ϕ_{jk} denotes the posterior probability $p(\gamma_j = k \,|\, x, z_j)$, and $L_{jk} \triangleq g_k(z_j)$. From this expression for the expected complete log-likelihood, the M-step update for the mixture weights works out to

$$x_k = \frac{1}{n} \sum_{j=1}^{n} \phi_{jk}, \tag{4.19}$$

and the E-step consists of computing the posterior probabilities,

$$\phi_{jk} = \frac{L_{jk} x_k}{\sum_{k'=1}^{m} L_{jk'} x_{k'}}. \tag{4.20}$$

In summary, the EM algorithm for solving (4.1) is easy to explain and implement: it iteratively updates the posterior probabilities for the current estimate of the mixture weights, following (4.20) (this is E-step), then updates the mixture weights according to (4.19) (this is the M-step), until some stopping criterion is met or until some upper limit on the number of iterations is reached.

4.6.2 Proofs

Here we provide proofs for Proposition 4.3.1 and Proposition 4.3.2.

Proof of Proposition 4.3.1

Because of the monotonicity and unboundedness of the objective over the positive orthant, $\mathbb{R}^m$, the solution to (4.8) is preserved if we relax the simplex constraint $\mathcal{S}^m = \{x : \mathbb{1}^T x = 1, x \succeq 0\}$ to a set of linear inequality constraints, $\{x : \mathbb{1}^T x \leq 1, x \succeq 0\}$. Slater's condition [22] is trivially satisfied for both formulations of the simplex constraints, and the feasible set is compact. The solution then

satisfies the KKT optimality conditions; *i.e.*, at the solution x^* there exists a $\lambda^* > 0$ and a $\mu^* \succeq 0$ such that

$$\begin{aligned} \nabla\phi(x^*) + \lambda^*\mathbb{1} - \mu^* &= 0 \\ (\mu^*)^T x^* &= 0 \\ \mathbb{1}^T x^* &= 1. \end{aligned} \tag{4.21}$$

We therefore conclude that solving (4.8) is equivalent to solving

$$\underset{x \in \mathbb{R}^m_+}{\text{minimize}}\ \phi(x) + \lambda^*\big(\textstyle\sum_{k=1}^m x_k - 1\big). \tag{4.22}$$

We claim that for any $\lambda > 0$, the solution to the Lagrangian relaxation of the problem,

$$x^*(\lambda) = \underset{x \in \mathbb{R}^m_+}{\operatorname{argmin}}\, \phi(x) + \lambda \textstyle\sum_{k=1}^m x_k,$$

is the same as solution to the original problem up to a constant of proportionality, and the proportionality constant is λ/λ^*. So long as $\lambda^* > 0$, we have that

$$\begin{aligned} x^*(\lambda) &= \underset{x \in \mathbb{R}^m_+}{\operatorname{argmin}} \big\{\phi(x) + \lambda \textstyle\sum_{k=1}^m x_k\big\} \\ &= \underset{x \in \mathbb{R}^m_+}{\operatorname{argmin}} \big\{\phi(\tfrac{\lambda}{\lambda^*}x) + \lambda^* \textstyle\sum_{k=1}^m \tfrac{\lambda}{\lambda^*} x_k\big\} \\ &= \frac{\lambda^*}{\lambda} \times \underset{x' \in \mathbb{R}^m_+}{\operatorname{argmin}} \big\{\phi(x') + \lambda^* \textstyle\sum_{k=1}^m x'_k\big\} \\ &= \tfrac{\lambda^*}{\lambda} x^*(\lambda^*). \end{aligned}$$

The second equality follows from the scale invariance assumption on $\phi(x)$.

Note that $\lambda^* = 0$ cannot hold at a solution to (4.8). Suppose that $\lambda^* = 0$. Then the point x^* must satisfy the KKT conditions of the problem in which the equality constraint $\mathbb{1}^T x = 1$ is removed, and it must then be a solution of that problem. Since the objective function decreases as we scale up x, such problems clearly are unbounded below and thus cannot have an optimal solution (as scaling

Algorithm 7 MIX-ACTIVE-SET: active set method method to compute a search direction for MIX-SQP.

Input: gradient, $g \in \mathbb{R}^m$; Hessian, $H \in \mathbb{R}^{m \times m}$;
initial working set, $\mathcal{W}^{(0)} \subseteq \{1, \ldots, m\}$;
convergence tolerance, $\epsilon \geq 0$.

Output: $y^{(l)} \in \mathbb{R}^m$, an estimate of the solution to (4.15). $k \leftarrow m - |W^{(0)}|$;

Set $y_i^{(0)} \leftarrow 0, \forall i \in \mathcal{W}^{(0)}$;
Set $y_i^{(0)} \leftarrow 1/k, \forall i \notin \mathcal{W}^{(0)}$;

```
for l = 0, 1, 2, ... do
  b ← Hy^(l) + 2g + 1_m;  (see eq. 4.16)
  q^(l) ← argmin_q q^T Hq/2 + q^T b s.t. q_i = 0, ∀i ∈ W^(l);  (see (4.16))
  α_l ← 1;
  if max_i |q_i^(l)| ≤ 0 then
    if min_{i ∈ W^(l)} b_i ≥ −ϵ then
      stop;  (all Lagrange multipliers in working set are positive)
      j ← argmin_{i ∈ W^(l)} b_i;
      W^(l+1) ← W^(l) \ {j};  (remove smallest multiplier from active set)
    else
      j ← argmin_{i ∉ W^(l), q_i^(l) < 0} −y_i^(l) / q_i^(l);
      α_l ← −y_j^(l) / q_j^(l); (find largest step size retaining feasibility)
      if α_l < 1 then
        W^(l+1) ← W^(l) ∪ {j};  (add blocking constraint to working set)
      end if
    end if
  end if
  y^(l+1) ← y^(l) + α_l q^(l)
end for
```

x up keeps decreasing the function value while preserving nonnegativity of entries in x). Since the solution of (4.8) must satisfy $\mathbb{1}^T x = 1$, the conclusion follows.

Proof of Proposition 4.3.2

The proof follows from the KKT optimality conditions (4.21). Premultiplying the first set of equations by $(x^\star)^T$, we have that

$$(x^*)^T \nabla f(x^*) + \lambda^* = 0.$$

The gradient of the objective in (4.1) is $\nabla f(x) = -L^T d/n$, where d is a vector of length n with entries $d_j = 1/(Lx)_j$. Inserting this expression for the gradient into the above identity yields $\lambda^* = 1$.

4.6.3 Implementation of the Active Set Method

At each iteration of MIX-SQP (Algorithm 6), an active set method is used to compute a search direction, $p^{(t)}$. The active set method is given in Algorithm 7. It is adapted from Algorithm 16.3 of [115]. Note that additional logic is needed to handle boundary conditions, such as the case when the working set is empty, and when the working set contains all co-ordinates except one.

4.6.4 GIANT Data Processing Details

We retrieved the file `GIANT_HEIGHT_Wood_et_al_2014_publicrelease_HapMapCeuFreq.txt.gz` from GIANT Project Wiki (`http://portals.broadinstitute.org/collaboration/giant`). The original tab-delimited text file contained summary statistics—regression coefficient estimates z_j and their standard errors s_j (columns "b" and "SE" in the text file)—for 2,550,858 SNPs j on chromosomes 1–22 and chromosome X. These summary statistics were computed from a meta-analysis of 79 genome-wide association studies of human height; see [167] for details about the studies and meta-analysis methods used. We filtered out 39,812 SNPs that were not identified in Phase 3 of the 1000 Genomes Project [7], an additional 384,254 SNPs where the coding strand was ambiguous, and 114 more SNPs with alleles that did not match the 1000 Genomes data, for a final data set containing $n = 2{,}126{,}678$ SNPs. (Note that the signs of the z_j estimates were flipped when necessary to align with the 1000 Genomes SNP genotype encodings, although this should have

had no effect on our results since the prior is a mixture of zero-centered normals.) The processed GIANT data are included in the accompanying source code repository.

REFERENCES

[1] Agarwal, N., Bullins, B., and Hazan, E. (2016). Second-order stochastic optimization for machine learning in linear time. *arXiv*, 1602.03943.

[2] Aharon, M., Elad, M., and Bruckstein, A. (2006). rmK-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322.

[3] Andersen, E. D. and Andersen, K. D. (2000). The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In *High performance optimization*, pages 197–232. Springer.

[4] Arnold, T. B. and Tibshirani, R. J. (2014). genlasso: Path algorithm for generalized lasso problems. *R package version*, 1.

[5] Atkinson, S. E. (1992). The performance of standard and hybrid EM algorithms for ML estimates of the normal mixture model with censoring. *Journal of Statistical Computation and Simulation*, 44(1–2):105–115.

[6] Attias, H. (1999). Independent factor analysis. *Neural Computation*, 11(4):803–851.

[7] Auton, A. (2015). A global reference for human genetic variation. *Nature*, 526(7571):68–74.

[8] Barron, A., Birgé, L., and Massart, P. (1999). Risk bounds for model selection via penalization. *Probability theory and related fields*, 113(3):301–413.

[9] Barry, D. and Hartigan, J. A. (1993). A Bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88(421):309–319.

[10] Beal, M. and Ghahramani, Z. (2003). The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. In Bernardo, J., Bayarri, M., Berger, J., Dawid, A., Heckerman, D., Smith, A., and West, M., editors, *Bayesian Statistics*, volume 7, pages 453–464. Oxford University Press.

[11] Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. Springer, New York, NY.

[12] Bertsekas, D. P. (1999). *Nonlinear Programming*. Athena Scientific, Belmont, MA, second edition.

[13] Bezanson, J., Karpinski, S., Shah, V. B., and Edelman, A. (2012). Julia: a fast dynamic language for technical computing. *arXiv*, 1209.5145.

[14] Bhattacharjee, A., Richards, W. G., Staunton, J., Li, C., Monti, S., Vasa, P., Ladd, C., Beheshti, J., Bueno, R., and Gillette, M. (2001). Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proceedings of the National Academy of Sciences*, 98(24):13790–13795.

[15] Birgin, E. G., Martínez, J. M., and Raydan, M. (2000). Nonmonotone spectral projected gradient methods on convex sets. *SIAM Journal on Optimization*, 10(4):1196–1211.

[16] Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Springer, New York, NY.

[17] Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: a review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

[18] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

[19] Bondell, H. D. and Reich, B. J. (2008). Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar. *Biometrics*, 64(1):115–123.

[20] Bottolo, L. and Richardson, S. (2010). Evolutionary stochastic search for Bayesian model exploration. *Bayesian Analysis*, 5(3):583–618.

[21] Bottou, L. and Le Cun, Y. (2004). Large scale online learning. In *Advances in Neural Information Processing Systems*, volume 16, pages 217–224.

[22] Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

[23] Breheny, P. and Huang, J. (2011). Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Annals of Applied Statistics*, 5(1):232–253.

[24] Brown, L. D. and Greenshtein, E. (2009). Nonparametric empirical Bayes and compound decision approaches to estimation of a high-dimensional vector of normal means. *Annals of Statistics*, pages 1685–1704.

[25] Burdakov, O. and Sysoev, O. (2017). A dual active-set algorithm for regularized monotonic regression. *Journal of Optimization Theory and Applications*, 172(3):929–949.

[26] Byrd, R. H., Hansen, S. L., Nocedal, J., and Singer, Y. (2016). A stochastic quasi-Newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031.

[27] Carbonetto, P. and Stephens, M. (2012). Scalable variational inference for Bayesian variable selection in regression, and its accuracy in genetic association studies. *Bayesian Analysis*, 7(1):73–108.

[28] Carvalho, C. M., Polson, N. G., and Scott, J. G. (2010). The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480.

[29] Cheng, Y. and Church, G. M. (2000). Biclustering of expression data. In *Ismb*, volume 8, pages 93–103.

[30] Chi, E. C., Allen, G. I., and Baraniuk, R. G. (2017). Convex biclustering. *Biometrics*, 73(1):10–19.

[31] Combettes, P. L. and Pesquet, J.-C. (2011). Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer.

[32] Cordy, C. B. and Thomas, D. R. (1997). Deconvolution of a distribution function. *Journal of the American Statistical Association*, 92(440):1459–1465.

[33] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977a). Maximum likelihood estimation from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38.

[34] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977b). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–22.

[35] Dharmadhikari, S. and Joag-Dev, K. (1988). *Unimodality, Convexity, and Applications*. Academic Press, Boston, MA.

[36] Diebolt, J. and Robert, C. P. (1994). Estimation of finite mixture distributions through Bayesian sampling. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 363–375.

[37] Drugowitsch, J. (2013). Variational Bayesian inference for linear and logistic regression. *arXiv*, 1310.5438.

[38] Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. (2008). Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pages 272–279. ACM.

[39] Dunning, I., Huchette, J., and Lubin, M. (2017). Jump: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320.

[40] Dykstra, R. L. (1983). An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842.

[41] Efron, B. (2008). Microarrays, empirical Bayes and the two-groups model. *Statistical Science*, 32(1):1–22.

[42] Efron, B. (2019). Bayes, oracle Bayes and empirical Bayes. *Statistical Science*, 34(2):177–201.

[43] 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.

[44] Fan, J. and Li, R. (2011). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360.

[45] Fan, Z. and Guan, L. (2018). Approximate ℓ_0-penalized estimation of piecewise-constant signals on graphs. *The Annals of Statistics*, 46(6B):3217–3245.

[46] Feige, U. and Krauthgamer, R. (2000). Finding and certifying a large hidden clique in a semirandom graph. *Random Structures & Algorithms*, 16(2):195–208.

[47] Friedman, J., Hastie, T., and Tibshirani, R. (2009). glmnet: Lasso and elastic-net regularized generalized linear models. *R package version*, 1(4).

[48] 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.

[49] Friedrich, F., Kempe, A., Liebscher, V., and Winkler, G. (2008). Complexity penalized m-estimation: Fast computation. *Journal of Computational and Graphical Statistics*, 17(1):201–224.

[50] Gamazon, E. R., Wheeler, H. E., Shah, K. P., Mozaffari, S. V., Aquino-Michaels, K., Carroll, R. J., Eyler, A. E., Denny, J. C., Nicolae, D. L., Cox, N. J., and Im, H. K. (2015). A gene-based association method for mapping traits using reference transcriptome data. *Nature Genetics*, 47(9):1091–1098.

[51] Gao, C., Han, F., and Zhang, C.-H. (2018). On estimation of isotonic piecewise constant signals. *The Annals of Statistics, to appear*.

[52] Gao, C., Lu, Y., Ma, Z., and Zhou, H. H. (2016). Optimal estimation and completion of matrices with biclustering structures. *The Journal of Machine Learning Research*, 17(1):5602–5630.

[53] Gao, C., van der Vaart, A. W., and Zhou, H. H. (2015). A general framework for bayes structured linear models. *arXiv preprint arXiv:1506.02174*.

[54] George, E. I. and Foster, D. P. (2000). Calibration and empirical Bayes variable selection. *Biometrika*, 87(4):731–747.

[55] George, E. I. and McCulloch, R. E. (1993). Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423):881–889.

[56] George, E. I. and McCulloch, R. E. (1997). Approaches for Bayesian variable selection. *Statistica Sinica*, 9(2):339–373.

[57] Gerard, D. and Stephens, M. (2020). Empirical Bayes shrinkage and false discovery rate estimation, allowing for unwanted variation. *Biostatistics*, 21(1):15–32.

[58] Ghahramani, Z. and Hinton, G. E. (2000). Variational learning for switching state-space models. *Neural Computation*, 12(4):831–864.

[59] Ghosh, A., Boyd, S., and Saberi, A. (2008). Minimizing effective resistance of a graph. *SIAM review*, 50(1):37–66.

[60] Ghosh, J. and Clyde, M. A. (2011). Rao–blackwellization for Bayesian variable selection and model averaging in linear and binary regression: A novel data augmentation approach. *Journal of the American Statistical Association*, 106(495):1041–1052.

[61] Girolami, M. (2001). A variational method for learning sparse and overcomplete representations. *Neural Computation*, 13(11):2517–2532.

[62] Golub, G. H. and Van Loan, C. F. (2012). *Matrix computations*, volume 3. JHU Press.

[63] Govaert, G. and Nadif, M. (2003). Clustering with block mixture models. *Pattern Recognition*, 36(2):463–473.

[64] GTEx Consortium (2017). Genetic effects on gene expression across human tissues. *Nature*, 550(7675):204–213.

[65] Guan, Y. and Stephens, M. (2011). Bayesian variable selection regression for genome-wide association studies and other large-scale problems. *Annals of Applied Statistics*, 5(3):1780–1815.

[66] Habier, D., Fernando, R. L., Kizilkaya, K., and Garrick, D. J. (2011). Extension of the Bayesian alphabet for genomic selection. *BMC Bioinformatics*, 12(1):186.

[67] Hajek, B., Wu, Y., and Xu, J. (2017). Submatrix localization via message passing. *The Journal of Machine Learning Research*, 18(1):6817–6868.

[68] Halko, N., Martinsson, P.-G., and Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288.

[69] Hallac, D., Leskovec, J., and Boyd, S. (2015). Network lasso: Clustering and optimization in large graphs. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 387–396. ACM.

[70] Hans, C., Dobra, A., and West, M. (2007). Shotgun stochastic search for "large p" regression. *Journal of the American Statistical Association*, 102(478):507–516.

[71] Hartigan, J. A. (1972). Direct clustering of a data matrix. *Journal of the american statistical association*, 67(337):123–129.

[72] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, NY, second edition.

[73] Hazimeh, H. and Mazumder, R. (2018). Fast best subset selection: coordinate descent and local combinatorial optimization algorithms. *arXiv*, 1803.01454.

[74] Ho, K. L. and Olver, S. (2018). LowRankApprox.jl: fast low-rank matrix approximation in Julia.

[75] Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

[76] Imrich, W. and Klavzar, S. (2000). *Product graphs: structure and recognition*. Wiley.

[77] Jaakkola, T. S. and Jordan, M. I. (2000). Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10(1):25–37.

[78] Javanmard, A. and Montanari, A. (2018). Debiasing the lasso: optimal sample size for Gaussian designs. *Annals of Statistics*, 46(6A):2593–2622.

[79] Jiang, W. and Zhang, C.-H. (2009). General maximum likelihood empirical bayes estimation of normal means. *Annals of Statistics*, 37(4):1647–1684.

[80] Johnstone, I. M. and Silverman, B. W. (2004a). Needles and straw in haystacks: empirical Bayes estimates of possibly sparse sequences. *Annals of Statistics*, 32(4):1594–1649.

[81] Johnstone, I. M. and Silverman, B. W. (2004b). Needles and straw in haystacks: empirical Bayes estimates of possibly sparse sequences. *Annals of Statistics*, 32(4):1594–1649.

[82] Johnstone, I. M. and Silverman, B. W. (2005). Empirical Bayes selection of wavelet thresholds. *Annals of Statistics*, pages 1700–1752.

[83] Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.

[84] Khatri, C. G. (1968). Some results for the singular normal multivariate regression models. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 267–280.

[85] Kiefer, J. and Wolfowitz, J. (1956). Consistency of the maximum likelihood estimator in the presence of infinitely many incidental parameters. *Annals of Mathematical Statistics*, pages 887–906.

[86] Kim, D., Sra, S., and Dhillon, I. S. (2010). Tackling box-constrained optimization via a new projected quasi-Newton approach. *SIAM Journal on Scientific Computing*, 32(6):3548–3563.

[87] Kim, Y., Carbonetto, P., Stephens, M., and Anitescu, M. (2020a). A fast algorithm for maximum likelihood estimation of mixture proportions using sequential quadratic programming. *Journal of Computational and Graphical Statistics, forthcoming*.

[88] Kim, Y. and Gao, C. (2019). Bayesian model selection with graph structured sparsity. *arXiv preprint arXiv:1902.03316*.

[89] Kim, Y., Wang, W., Carbonetto, P., and Stephens, M. (2020b). A fast and flexible empirical Bayes approach for prediction in multiple regression. *in preparation*.

[90] Koenker, R. and Gu, J. (2017). REBayes: an R package for empirical Bayes mixture methods. *Journal of Statistical Software*, 82(8):1–26.

[91] Koenker, R. and Mizera, I. (2014). Convex optimization, shape constraints, compound decisions, and empirical Bayes rules. *Journal of the American Statistical Association*, 109(506):674–685.

[92] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86.

[93] Laird, N. (1978). Nonparametric maximum likelihood estimation of a mixing distribution. *Journal of the American Statistical Association*, 73(364):805–811.

[94] Lee, M., Shen, H., Huang, J. Z., and Marron, J. (2010). Biclustering via sparse singular value decomposition. *Biometrics*, 66(4):1087–1095.

[95] Lehmann, E. L. and Casella, G. (1998). *Theory of Point Estimation*. Springer, New York, NY, second edition.

[96] Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., and Ghahramani, Z. (2010). Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, 11(Feb):985–1042.

[97] Li, F. and Zhang, N. R. (2010). Bayesian variable selection in structured high-dimensional covariate spaces with applications in genomics. *Journal of the American statistical association*, 105(491):1202–1214.

[98] Livne, O. E. and Brandt, A. (2012). Lean algebraic multigrid (lamg): Fast graph laplacian linear solver. *SIAM Journal on Scientific Computing*, 34(4):B499–B522.

[99] Logsdon, B. A., Hoffman, G. E., and Mezey, J. G. (2010). A variational Bayes algorithm for fast and accurate multiple locus genome-wide association analysis. *BMC Bioinformatics*, 11(1):58.

[100] Lovász, L. (1993). Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46.

[101] Lu, M. and Stephens, M. (2016). Variance adaptive shrinkage (vash): flexible empirical Bayes estimation of variances. *Bioinformatics*, 32(22):3428–3434.

[102] Ma, Z. and Wu, Y. (2015). Volume ratio, sparsity, and minimaxity under unitarily invariant norms. *IEEE Transactions on Information Theory*, 61(12):6939–6956.

[103] MacArthur, J., Bowler, E., Cerezo, M., Gil, L., Hall, P., Hastings, E., Junkins, H., McMahon, A., Milano, A., Morales, J., Pendlington, Z. M., Welter, D., Burdett, T., Hindorff, L., Flicek, P., Cunningham, F., and Parkinson, H. (2017). The new NHGRI-EBI catalog of published genome-wide association studies (GWAS Catalog). *Nucleic Acids Research*, 45(D1):D896–D901.

[104] Mair, P., Hornik, K., and de Leeuw, J. (2009). Isotone optimization in r: pool-adjacent-violators algorithm (pava) and active set methods. *Journal of statistical software*, 32(5):1–24.

[105] McCullagh, P. and Nelder, J. A. (1989). *Generalized Linear Models*, volume 37 of *Monographs on Statistics and Applied Probability*. Chapman and Hall, New York, NY, second edition.

[106] Meuwissen, T. H., Hayes, B. J., and Goddard, M. E. (2001). Prediction of total genetic value using genome-wide dense marker maps. *Genetics*, 157(4):1819–1829.

[107] Meuwissen, T. H. E., Solberg, T. R., Shepherd, R., and Woolliams, J. A. (2009). A fast algorithm for BayesB type of prediction of genome-wide estimates of genetic value. *Genetics Selection Evolution*, 41:2.

[108] Miller, A. J. (2002). *Subset Selection in Regression*. Chapman and Hall/CRC.

[109] Mitchell, T. J. and Beauchamp, J. J. (1988). Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032.

[110] Morris, C. N. (1983). Parametric empirical Bayes inference: theory and applications. *Journal of the American Statistical Association*, 78(381):47–55.

[111] Moser, G., Lee, S. H., Hayes, B. J., Goddard, M. E., Wray, N. R., and Visscher, P. M. (2015). Simultaneous discovery, estimation and prediction analysis of complex traits using a Bayesian mixture model. *PLOS Genetics*, 11(4):e1004969.

[112] Neal, R. M. and Hinton, G. E. (1998a). A new view of the EM algorithm that justifies incremental, sparse, and other variants. In Jordan, M. I., editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers.

[113] Neal, R. M. and Hinton, G. E. (1998b). A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer.

[114] Nebebe, F. and Stroud, T. (1986). Bayes and empirical Bayes shrinkage estimation of regression coefficients. *Canadian Journal of Statistics*, 14(4):267–280.

[115] Nocedal, J. and Wright, S. J. (2006). *Nonlinear Optimization*. Springer, New York, NY.

[116] Parikh, N. and Boyd, S. (2014). Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239.

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

[118] Parmigiani, G. and Inoue, L. (2009). *Decision theory: Principles and approaches*, volume 812. John Wiley & Sons.

[119] Pearson, K. (1894). Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 185:71–110.

[120] Pérez, P. and de Los Campos, G. (2014). Genome-wide regression and prediction with the BGLR statistical package. *Genetics*, 198(2):483–495.

[121] Potra, F. A. and Wright, S. J. (2000). Interior-point methods. *Journal of Computational and Applied Mathematics*, 124(1–2):281–302.

[122] R Core Team (2017). *R: a Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

[123] Ray, K. and Szabo, B. (2019). Variational Bayes for high-dimensional linear regression with sparse priors. *arXiv preprint arXiv:1904.07150*.

[124] Redner, R. A. and Walker, H. F. (1984). Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239.

[125] Richardson, S. and Green, P. J. (1997). On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society: series B (statistical methodology)*, 59(4):731–792.

[126] Robbins, H. (1964). The empirical Bayes approach to statistical decision problems. *Annals of Mathematical Statistics*, 35(1):1–20.

[127] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407.

[128] Robert, C. and Casella, G. (2013). *Monte Carlo statistical methods*. Springer Science & Business Media.

[129] Rockova, V. (2013). Bayesian variable selection in high-dimensional applications.

[130] Ročková, V. and George, E. I. (2014). Emvs: The em approach to Bayesian variable selection. *Journal of the American Statistical Association*, 109(506):828–846.

[131] Ročková, V. and George, E. I. (2018). The spike-and-slab lasso. *Journal of the American Statistical Association*, 113(521):431–444.

[132] Salakhutdinov, R., Roweis, S., and Ghahramani, Z. (2003). Optimization with EM and expectation-conjugate-gradient. In *Proceedings of the 20th International Conference on Machine Learning*, pages 672–679.

[133] Saul, L. K. and Jordan, M. I. (1996). Exploiting tractable substructures in intractable networks. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems 8*, pages 486–492. MIT Press.

[134] Schell, M. J. and Singh, B. (1997). The reduced monotonic regression method. *Journal of the American Statistical Association*, 92(437):128–135.

[135] Schmidt, M., Berg, E., Friedlander, M., and Murphy, K. (2009). Optimizing costly functions with simple constraints: a limited-memory projected quasi-Newton algorithm. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 456–463.

[136] She, Y. (2010). Sparse regression with exact clustering. *Electronic Journal of Statistics*, 4:1055–1096.

[137] Sill, M., Kaiser, S., Benner, A., and Kopp-Schneider, A. (2011). Robust biclustering by sparse singular value decomposition incorporating stability selection. *Bioinformatics*, 27(15):2089–2097.

[138] Smyth, G. K. (2004). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 3(1).

[139] Song, Q. and Cheng, G. (2018). Optimal false discovery control of minimax estimator. *arXiv preprint arXiv:1812.10013*.

[140] Songgui, W. and Shein-Chung, C. (1994). Advanced linear models: Theory and applications.

[141] Spielman, D. A. (2007). Spectral graph theory and its applications. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pages 29–38. IEEE.

[142] Spielman, D. A. and Teng, S.-H. (2004). Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 81–90. ACM.

[143] Stephens, M. (2016a). False discovery rates: a new deal. *Biostatistics*, 18(2):275–294.

[144] Stephens, M. (2016b). False discovery rates: a new deal. *Biostatistics*, 18(2):275–294.

[145] Stephens, M., Carbonetto, P., Dai, C., Gerard, D., Lu, M., Sun, L., Willwerscheid, J., Xiao, N., and Zeng, M. (2018). *ashr: Methods for Adaptive Shrinkage, using Empirical Bayes*.

[146] Stigler, S. M. (1984). Studies in the history of probability and statistics XL: Boscovich, Simpson and a 1760 manuscript note on fitting a linear relation. *Biometrika*, 71(3):615–620.

[147] Su, W., Bogdan, M., and Candes, E. (2017). False discoveries occur early on the lasso path. *Annals of Statistics*, 45(5):2133–2150.

[148] Sun, L. and Stephens, M. (2018). Solving the empirical bayes normal means problem with correlated noise. *arXiv preprint arXiv:1812.07488*.

[149] Tan, K. M. and Witten, D. M. (2014). Sparse biclustering of transposable data. *Journal of Computational and Graphical Statistics*, 23(4):985–1008.

[150] Tao, T. (2012). *Topics in random matrix theory*, volume 132. American Mathematical Soc.

[151] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

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

[153] Tibshirani, R. J., Hoefling, H., and Tibshirani, R. (2011). Nearly-isotonic regression. *Technometrics*, 53(1):54–61.

[154] Tibshirani, R. J. and Taylor, J. (2011a). The solution path of the generalized lasso. *The Annals of Statistics*, 39(3):1335–1371.

[155] Tibshirani, R. J. and Taylor, J. (2011b). The solution path of the generalized lasso. *The Annals of Statistics*, 39(3):1335–1371.

[156] Tikhonov, A. N. (1963). Solution of incorrectly formulated problems and the regularization method. *Soviet Mathematics—Doklady*, 501–504.

[157] Urbut, S. M., Wang, G., Carbonetto, P., and Stephens, M. (2019). Flexible statistical methods for estimating and testing effects in genomic studies with multiple conditions. *Nature Genetics*, 51(1):187–195.

[158] Varadhan, R. and Roland, C. (2008). Simple and globally convergent methods for accelerating the convergence of any EM algorithm. *Scandinavian Journal of Statistics*, 35(2):335–353.

[159] Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305.

[160] Wang, B. and Titterington, D. (2005). Inadequacy of interval estimates corresponding to variational Bayesian approximations. In *AISTATS*. Citeseer.

[161] Wang, C. and Blei, D. M. (2018). A general method for robust Bayesian modeling. *Bayesian Analysis*, 13(4):1163–1191.

[162] Wang, G., Sarkar, A., Carbonetto, P., and Stephens, M. (2019a). A simple new approach to variable selection in regression, with application to genetic fine-mapping. *bioRxiv*, doi:10.1101/2019.12.11.123456.

[163] Wang, W. (2017). *Applications of Adaptive Shrinkage in Multiple Statistical Problems*. The University of Chicago.

[164] Wang, W. and Stephens, M. (2018). Empirical Bayes Matrix Factorization. *arXiv preprint arXiv:1802.06931*.

[165] Wang, Y., Miller, A. C., and Blei, D. M. (2019b). Comment: Variational autoencoders as empirical Bayes. *Statistical Science*, 34(2):229–233.

[166] Witten, D. M. and Tibshirani, R. (2010). A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490):713–726.

[167] Wood, A. R., Esko, T., Yang, J., Vedantam, S., and Pers, T. H. (2014). Defining the role of common variation in the genomic and biological architecture of adult human height. *Nature Genetics*, 46(11):1173—-1186.

[168] Wright, S. J. (1998). Superlinear convergence of a stabilized sqp method to a degenerate solution. *Computational Optimization and Applications*, 11(3):253–275.

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

[170] Wu, W. B., Woodroofe, M., and Mentz, G. (2001). Isotonic regression: Another look at the changepoint problem. *Biometrika*, 88(3):793–804.

[171] Xing, Z., Carbonetto, P., and Stephens, M. (2016). Flexible signal denoising via flexible empirical Bayes shrinkage. *arXiv preprint arXiv:1605.07787*.

[172] Xu, L. and Jordan, M. I. (1996). On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Computation*, 8(1):129–151.

[173] Xu, S. and Fan, Z. (2019). Iterative alpha expansion for estimating gradient-sparse signals from linear measurements. *arXiv preprint arXiv:1905.06097*.

[174] Yuan, M. and Lin, Y. (2005). Efficient empirical Bayes variable selection and estimation in linear models. *Journal of the American Statistical Association*, 100(472):1215–1225.

[175] Zeng, J., de Vlaming, R., Wu, Y., Robinson, M. R., Lloyd-Jones, L. R., Yengo, L., Yap, C. X., Xue, A., Sidorenko, J., McRae, A. F., Powell, J. E., Montgomery, G. W., Metspalu, A., Esko, T., Gibson, G., Wray, N. R., Visscher, P. M., and Yang, J. (2018). Signatures of negative selection in the genetic architecture of human complex traits. *Nature Genetics*, 50(5):746–753.

[176] Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942.

[177] Zhang, C.-H., Zhang, T., et al. (2012). A general theory of concave regularization for high-dimensional sparse estimation problems. *Statistical Science*, 27(4):576–593.

[178] Zhou, H. and Zheng, T. (2013). Bayesian hierarchical graph-structured model for pathway analysis using gene expression data. *Statistical applications in genetics and molecular biology*, 12(3):393–412.

[179] Zhou, X., Carbonetto, P., and Stephens, M. (2013). Polygenic modeling with Bayesian sparse linear mixed models. *PLOS genetics*, 9(2).

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