Topics in High-Dimensional Statistics and the Analysis of Large Hyperspectral Images

by

Chia Chye Yee

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in the University of Michigan
2016

Doctoral Committee:

Professor Yves A. Atchadé, Chair
Professor Kerby A. Shedden
Professor Ambuj Tewari
Professor Xiaoquan William Wen


UNIVERSITY OF
MICHIGAN

This dissertation is in honor of my parents, without whom this would not be possible. I am eternally grateful and indebted to your kindness and sacrifice in making all this possible.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my thesis advisor Professor Yves Atchadé for his guidance and support in my graduate studies. The frequent meetings and discussions about research topics has broadened my horizons as a researcher. He is always patient and understanding in answering all my potentially silly questions. In addition, his willingness to share his insights and expertise has been crucial to my academic development and the completion of my graduate study.

I would like to thank Professor Kerby Shedden for giving me the opportunity to work at Consulting for Statistics, Computing and Analytic Research (CSCAR) as a graduate student consultant. Working there has exposed me to practical statistical concerns faced by researchers and gave me many cherished memories.

I would like to thank all committee members for volunteering their time and effort in evaluating my thesis. Some of the work/teaching by the committee members has provided me with a source of inspiration/reference in my everyday research.

I would like to thank the excellent administrative staff at the Statistics Department. Their help has been invaluable in helping me navigate the countless bureaucratic red tape throughout my time as a graduate student. Without them, I would be completely lost and the department would be a less cheerful place.

I would also like to thank my classmates in Michigan for being great friends and excellent lunch-mates. Our weekly lunches has provided me with some healthy light-hearted discussion and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

ADMM alternating direction method of multipliers

BIC Bayesian information criterion

EM expectation-maximization

LASSO least absolute shrinkage and selection operator

MCMC Monte Carlo Markov Chain

MLE maximum likelihood estimate

SBL sparse Bayesian learning

SVM support vector machine

USGS United States Geological Survey

VCA vertex component analysis

ABSTRACT

Advancement in imaging technology has made hyperspectral images gathered from remote sensing much more common. The high-dimensional nature of these large scale data coupled with wavelength and spatial dependency necessitates high-dimensional and efficient computation methods to address these issues while producing results that are concise and easy to understand. The thesis addresses these issues by examining high-dimensional methods in the context of hyperspectral image classification, unmixing and wavelength correlation estimation.

Chapter 2 re-examines the sparse Bayesian learning (SBL) of linear models of [88] in a high-dimensional setting with sparse signal. The hard-thresholded version of the SBL estimator, under orthogonal design, achieves non-asymptotic error rate that is comparable to LASSO of [87]. We also establish in the chapter that with high-probability the estimator recovers the sparsity structure of the signal. The ability to recover sparsity structures in high dimensional settings is crucial for unmixing with high-dimensional libraries in the next chapter. In **Chapter 3**, the thesis investigates the application of SBL on the task of linear/bilinear unmixing and classification of hyperspectral images. The proposed model in this chapter uses latent Markov random fields to classify pixels and account for the spatial dependence between pixels. In the proposed model, the pixels belonging to the same group share the same mixture of pure endmembers. The task of unmixing and classification

are performed simultaneously, but this method does not address wavelength dependence. **Chapter 4** is a natural extension of the previous chapter that contains the framework to account for both spatial and wavelength dependence in the unmixing of hyperspectral images. The classification of the images are performed using approximate spectral clustering while the unmixing task is performed in tandem with sparse wavelength concentration matrix estimation.

CHAPTER 1

Introduction

The past decade in statistics has been defined by the exponential growth of data and computational power. However, abundance of data in the absence of insight is sometimes more detrimental than the absence of data because no inference is better than wrong/spurious inference. In addition, the pace of data growth shows no sign of slowing down as the data grows ever more complex. The increase in ability to store large amounts of high-dimensional data coupled with diminishing cost involved in such endeavors necessitate principled and structured approaches to draw inference and gain insight from data.

The increase in complexity of the data gathered means the number of parameters very often far exceeds the number of samples. Under such conditions, classical solutions to problems such as the ordinary least squares are often sub-optimal and inconsistent. As a result, research into shrinkage estimators such as LASSO [84, 87] (and its consistency in model selection [22, 61, 90, 103]) and its derivative elastic net [104] have been introduced to induce sparse structures in the solution to high-dimensional models. The methods mentioned thus far are frequentist methods. At the other end of the spectrum, Bayesian methods involve regularization via prior formulation [52, 71] with Gaussian cases particularly well understood [94]. On the the other hand, there is also some work on empirical Bayes calibration, computation and variable selection as documented in [8, 24, 25, 41, 55, 62].

High dimensional methods are particularly relevant in the field of hyperspectral data processing. The rest of the chapter is an introduction to high-dimensional methods in the field of hyperspectral data processing. **Section 1.1** is an introduction to an empirical Bayes method that is useful in hyperpectral modeling. **Section 1.2** is an overview of hyperspectral data, models, and application of aforementioned empirical Bayes method in hyperspectral classification and unmixing. **Section 1.3** is an introduction to the extension of the hyperspectral model to account for wavelength dependence.

1.1 The SBL in High-Dimensional Settings

As noted earlier, high-dimensional variable selection has become an important topic in statistics due to increase in data complexity. The LASSO [87] is the most widely used method in tackling this problem. Although this method has proven to be successful, there are issues regarding the right level of regularization. Choosing the appropriate level of penalization remains a computer-intensive issue for many models. In addition to the frequentist approach, there is the Bayesian variable selection approach [16, 26, 53, 71, 73]. The Bayesian approach to this problem often generates intractable posteriors that require extensive MCMC simulations. In the middle of the frequentist and Bayesian spectrum, lies the empirical Bayes alternative known as sparse Bayesian learning (SBL) [37, 88, 96, 97] which is less popular. **Chapter 2** contains a reexamination of an empirical Bayes approach to penalized regression. The method examined in this chapter has the advantage of setting penalization level automatically via setting the hyperparameters to the maximum likelihood estimate (MLE). The method as outlined in [88] is inconsistent and only selects the true zero for each variable about 70% of the time. Even though the estimator as originally formulated does not recover the true zeros, many of the actual zeros are estimated as close to zero. We proposed an empirical threshold and show that the thresholded estimator that is consistent in terms of estimation error and model selection for the case with orthogonal design. With high probability, we show that the estimation error is of the same order as the LASSO and the sparsity structure is recovered provided the signal is not too weak. According to the simulation study comparing the method to LASSO, SBL outperforms the LASSO if the signal is sufficiently strong. However, under weak signal conditions, LASSO outperforms SBL.

1.2 Unmixing and Classification of Hyperspectral Images

Structure can also come in the form of parsimony via classification. The complex task of drawing inference from image based data is often simplified by classification which brings us to the focus of **Chapter 3** and **Chapter 4** of this thesis: hyperspectral images. A regular image is a collection of pixels with each pixel containing readings on the visible spectrum (typically 0.4μm to 0.7μm). In contrast, a hyperspectral image is a collection of pixels with each pixel containing readings on a very wide spectrum (typically 0.4μm to 2.5μm) [82]. In other words, hyperspectral images are more saturated in terms of wavelength bands. **Figure 1.1** shows an illustrated example of what consist of a hyperspectral image. The high spectral resolution captured in each pixel enables what is commonly referred to as "unmixing"

in the field. In hyperspectral images, due to the limited spatial resolution of the sensors, a pixel often contains a mixture of a few endmember ("pure" material) spectral signatures. Hyperspectral unmixing is the task of decomposing a pixel into its constituent endmembers [82]. Due to recent advances remote sensing technology, large (as large as several megapixels) hyperspectral images with high spectral resolutions have become ubiquitous. This opens up more possibilities in terms of application of this class of data in forest conservation, resource management etc. Presenting the results of the unmixing at the pixel level leads to information overload and is counter productive. This is where classification of pixels comes in handy. Typically pixels belonging to the same class share certain common characteristics that can be modeled parsimoniously and concisely represented. The unmixing of hyperspectral images, however, is complicated by the atmospheric attenuation of the signal received by the sensor and the angle at which the electromagnetic radiation reaches the sensor [82]. The problem of atmospheric attenuation would not be an issue in the case of medical hyperspectral images [57], but the modeling considerations remain the same (ie. unmixing and classification). The hyperspectral data analyzed in this thesis document covers mainly remote sensing data coming from satellites.

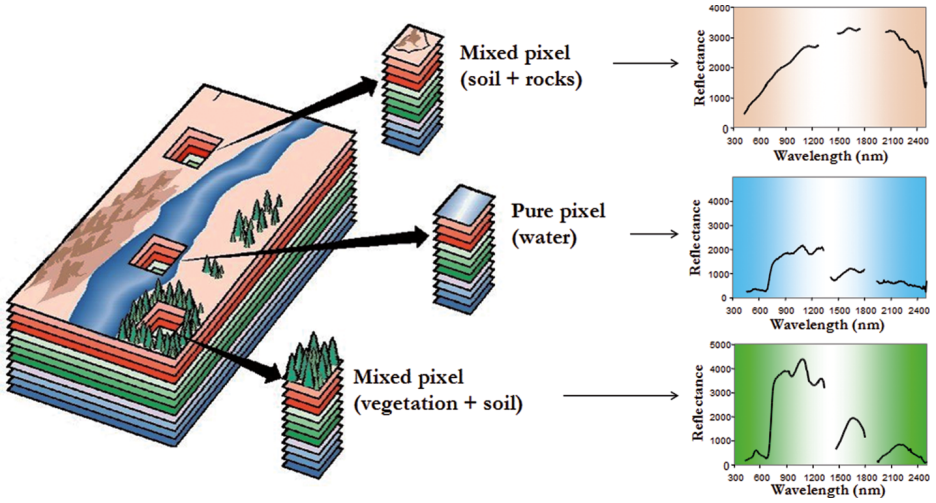


Figure 1.1: An illustrated example of a hyperspectral scene

An endmember is the spectral signature of a material of interest and an unmixing "library" is a set of endmembers compiled into a matrix. The unmixing of the hyperspectral images requires a library of reference spectral signatures (endmembers) which can either be lab-generated or spectral signatures extracted from the images. Either library has its own sets of assumptions and shortcomings. The lab-generated libraries [11] assume that the atmospheric distortion is minimal and does not drastically alter the spectral signature

of endmembers. These kind of libraries may require some on-site calibration [30] which is potentially difficult and expensive. In addition, these libraries are usually high-dimensional because there are usually more endmembers than spectral bands (spectral resolution) in the library. These high-dimensional libraries require the use of regularized methods such as LASSO [87] in order to recover a mixture that contains relatively few endmembers. Other than LASSO, there are other more exotic regularization such as the Laplacian regularization [7], sparsity promoting priors [102], and spatial LASSO regularization [81]. On the other hand, the extraction of spectral signatures from the images to form a library assumes the existence of a "pure" pixel in the image. Spectral extraction methods involve the solving for the library in addition to the mixture of the endmembers which complicates the task of unmixing further. In addition, this technique necessitates an additional post-processing step of identifying the spectral signatures extracted from the image via comparison.

Mixing models can be either linear or nonlinear, with the linear model being commonly used. **Figure 1.2** shows an illustration of the linear and nonlinear mixing of endmembers in a pixel. Although there are microscopic mixtures and methods to unmix them [19, 29, 65] (**Figure 1.2 (b)**), the focus of nonlinear mixing in this document is primarily on the bilinear model [47]. When we mention nonlinear mixing in this document, we generally refer to mixing examples falling in category **Figure 1.2 (c)** unless explicitly stated otherwise. Nonlinear mixing appears when there are macroscopic interactions of spectral signatures, which is most common to scenes with multi-layered configurations. In this type of scenes, the light scattered by a material is reflected off another material as seen in **Figure 1.2 (c)**. This usually happens in scenes with forest canopy where light scattered by foliage is reflected off the ground. Bilinear unmixing models are commonly used for this type of images. [34] and the references therein are a good starting point for further details on nonlinear unmixing.

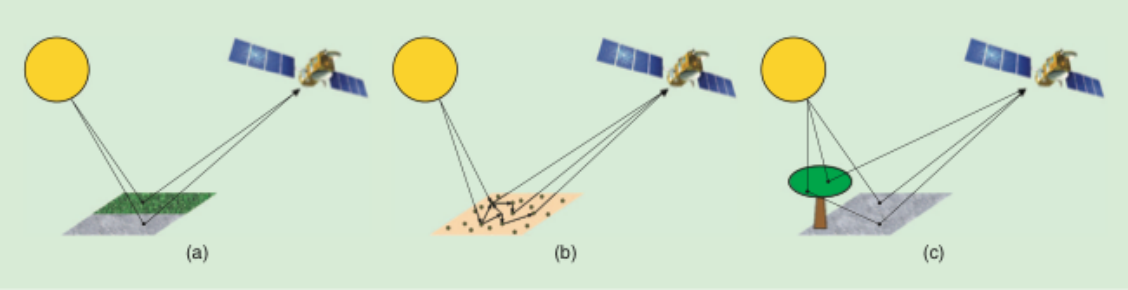


Figure 1.2: An illustrated example of linear and nonlinear mixing schemes [34]: (a) Linear mixing model: pixel contains linear mixture of two materials, (b) Intimate mixture: pixel contains microscopic mixture of several materials, (c) Bilinear model: pixel contains two endmembers, tree and soil.

Besides unmixing, there is also significant work on the classification of hyperspectral images. These methods are used in conjunction with unmixing to present data in manageable chunks. It is typical that the pixels sharing the same class also have similar characteristics. For instance in [89], the pixels belonging to the same class share the same first and second moment of abundances (mixtures) while [6] presents a model where the sparse abundances in the local neighborhood are correlated a priori. In addition to simplifying the data generative model, classification of images also have the advantage of introducing spatial dependence between the pixels: pixels belonging to the same class are more spatially correlated to each other relative to pixels belonging to another class. Due to the utility of image classification, it is no surprise that classification of hyperspectral images is an active area of research. Spatial dependence modeling is usually introduced into the unmixing model via Markov random fields [64]. Empirical methods such as support vector machine (SVM) [63, 85] have been applied on observed spectral signatures in order to separate pixels into classes. Besides that, there are Bayesian methods which assume that the mixture of endmembers in the same group share the same first and second moment in a Markov random field framework [89]. For a review of methods used in hyperspectral image classification, [3, 21] and the references therein are good resources. For alternative spatial dependence models utilizing the distance between the abundances belonging to neighboring pixels, [51, 92] are good references.

In most hyperspectral literature, the task of umixing and classification are implemented in a two-stage process. In **Chapter 3**, we examine a framework in which both these tasks are performed simultaneously. Under the framework outlined, the library used can either be lab-based or generated from spectral extraction methods such as N-FINDR [76, 95],

VCA [66], and Pixel Purity Index [15]. In addition, the columns of the library used can also be either linear or nonlinear. Spatial dependence between the pixels is modeled using Markov random fields. The model outlined builds on the Bayesian framework outlined in [89] with a notable difference: our model is more parsimonious in terms of the number of parameters. This enables the use of much higher-dimensional libraries. Besides that, we adopt the SBL [88] approach in **Chapter 2** in order to recover sparse solutions for the mixtures.

1.3 Hyperspectral Unmixing with Wavelength Dependence

Extensions of sparse linear methods naturally lead into penalized covariance/concentration [54, 74, 78] estimation. Typically these estimates are constructed around finding the penalized MLE of the Gaussian likelihood. Under the classical setting where the number of samples far exceeds the number of parameters, the MLE of the covariance matrix under the Gaussian likelihood is the sample covariance matrix. However, in the high-dimensional case, the MLE is not well behaved hence the interest in the penalized MLE. The penalized MLE for covariance/concentration matrix seeks to set a large number of off-diagonals to zero [4, 75]. An off-diagonal zero entry of the covariance matrix for Gaussian data implies marginal independence for the two respective variables while an off-diagonal zero entry for the concentration matrix under similar condition implies conditional independence. The independence implied by the respective sparsity structures is important in the general theme of imposing structure on an abundance of data.

To the best of our knowledge, there is no existing method that incorporates wavelength dependence in modeling hyperspectral unmixing. In **Chapter 4** we try to address both spatial and spectral wavelength dependence in the unmixing of hyperspectral data. **Figure 1.3** shows some example spectral signatures of common endmembers in hyperspectral images. Note the smooth curves exhibited by the spectral signatures. If there is no wavelength dependence, the curves would be more erratic and jagged. Empirical evidence suggests, the contiguous wavelength bands are likely to be correlated with each other if there is no abnormal/significant absorbtion in contiguous bands. This motivates the inclusion of wavelength dependence in our modeling considerations. We modeled the wavelength dependence explicitly via the concentration (inverse of the covariance) matrix in the multivariate Gaussian. As noted in **Section B.1**, a zero entry in the off-diagonal of the Gaussian precision matrix implies conditional independence of the wavelength bands. Therefore, we intend to recover sparse concentration (sparse off-diagonal) matrix to only account for the most correlated/dependent wavelength bands. We implement a penalized likelihood approach in

estimating the sparse concentration matrix.

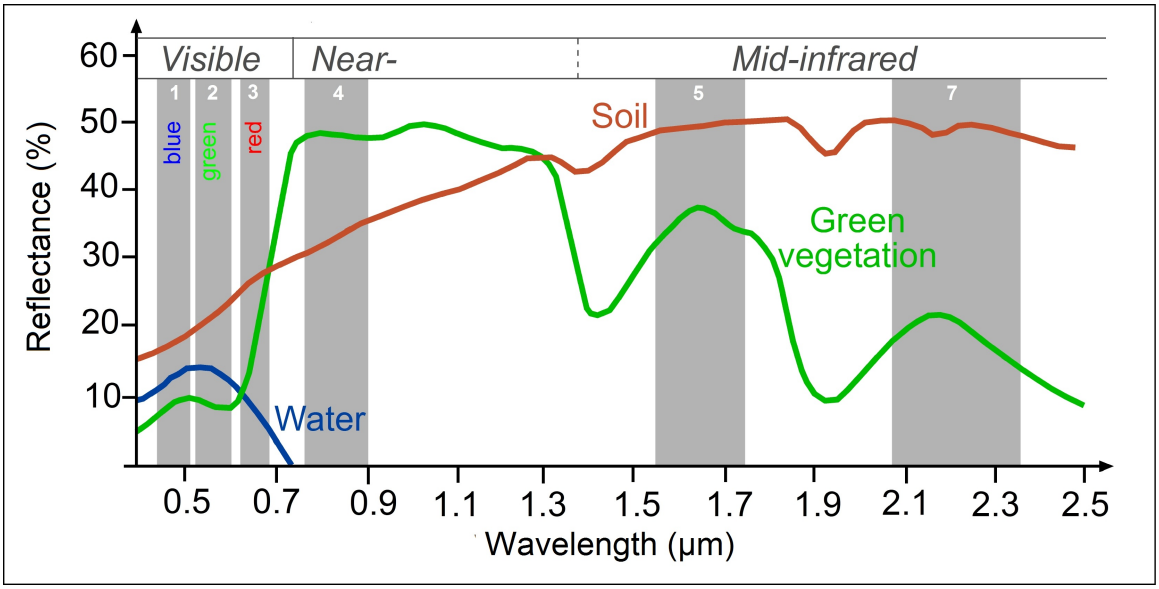


Figure 1.3: Examples of spectral signatures of endmembers such as vegetation, soil, and water within wavelength bands ranging from 0.4 μm to 2.5 μm

In addition to solving for the concentration matrix, this chapter also covers the constrained optimization problem of solving for the abundances (unmixing) of endmembers. The abundances are constrained to sum up to one and must be positive. While the wavelength dependence is explicitly modeled as a parameter in this model, spatial dependence of the pixels is implicitly modeled via classification of the pixels using spectral clustering. Spectral clustering requires the use of affinity/adjacency/weighting matrix with a distance based kernel [58]. The main idea of using spectral clustering is to group pixels based on the relative spectral signature distance to each other. In our implementation, we used the euclidean distance in the kernel. Although empirical evidence suggests this method is extremely effective, the memory and computation constraints of spectral clustering limits its use to small hyperspectral images. Parallel spectral clustering can be done by performing spectral clustering on small partitions of the hyperspectral image. The approach we take to tackling this problem involves approximate spectral clustering using the Nystrom method. Accounting for spatial and wavelength dependence in this model necessitates a two-stage method where the pixels are classified before unmixing with spatial dependence is performed within each class. Results from the real data simulation are compared to alternative methods using BIC and extended BIC.

CHAPTER 2

On the Sparse Bayesian Learning of Linear Models

2.1 Introduction

High-dimensional variable selection has become an important topic in modern statistics. The LASSO of [87] is probably the most widely used method for this problem and has span an extensive literature (see e.g. the monograph [20]). Despite its success, the method has many shortcomings. For instance choosing the right amount of regularization remains a difficult and computer-intensive issue for many models. In parallel to the frequentist approach, Bayesian variable selection for high-dimensional problems has also generated a large literature (see for instance [16, 26, 53, 71, 73] and the reference therein). But most Bayesian variable selection methods often lead to intractable posterior distributions that require a heavy use of Markov Chain Monte Carlo simulation. Between these two well-established frameworks lies an empirical Bayes alternative known as SBL [37, 88, 96, 97], which has received much less attention in the statistics literature.

This chapter is a re-examination of the SBL for linear regression in a high-dimensional setting. An interesting question is whether the SBL procedure recovers the sparsity structure of underlying signals. This problem was considered by [96] which establishes that in the noiseless setting the SBL indeed recovers the sparsity structure of the regression coefficients. However the method behaves differently in a noisy setting. For orthogonal design matrices, we show that the SBL indeed produces a sparse solution of the regression coefficients, but does not in general recover the sparsity structure of the regression coefficients. To remedy this limitation we propose a hard-thresholded version of the SBL estimator. We show that with high probability the thresholded estimator achieves the same estimation error of $O(\sigma\sqrt{s\log(p)/n})$ as LASSO, where n is the sample size, σ is the regression model standard deviation, p the number of regressors and s the number of non-zero regression coefficients. Furthermore we show that with high probability this thresholded estimator

recovers the sparsity structure of the regression coefficients provided that the signal is not too weak.

Finally we did a simulation study comparing SBL, thresholded SBL, and LASSO. We find that the performance of the thresholded SBL depends on the strength of the signal (defined here as the minimum of the absolute value of the non-zero coefficients). With a weak signal the thresholded SBL performs poorly compared to LASSO, but outperforms LASSO when the signal is strong.

2.2 Sparse Bayesian learning of linear regression models

Suppose that we observe a vector $y \in \mathbb{R}^n$ that is a realization of a random variable Y such that

$$Y = X\beta_\star + \epsilon, \tag{2.1}$$

for a known and non-random design matrix $X \in \mathbb{R}^{n\times p}$, a vector $\beta_\star \in \mathbb{R}^p$, and a random error term $\epsilon \in \mathbb{R}^n$ such that

$$\mathbb{E}(\epsilon) = 0, \quad \text{and} \quad \mathbb{E}(\epsilon\epsilon') = \sigma_\star^2 \mathbb{I}_n, \tag{2.2}$$

for $\sigma_\star^2 > 0$, where $\mathbb{I}_n$ is the n-dimensional identity matrix. Our objective is to estimate $\beta_\star$ and $\sigma_\star^2$. Although (2.1-2.2) does not make any specific distributional assumption on Y, we will consider the following possibly misspecified model: $Y \sim \mathbf{N}(X\beta, \sigma^2 I_n)$, with parameter $(\beta, \sigma^2) \in \mathbb{R}^p \times (0,\infty)$, where $\mathbf{N}(\mu, \Sigma)$ denotes the Gaussian distribution with mean μ and covariance matrix Σ. The parameter σ^2 is taken as fixed, and we assign to β a prior distribution of the form

$$\pi_\gamma(\mathrm{d}\beta) \stackrel{\text{def}}{=} \prod_{j=1}^{p} p_{\gamma_j}(\mathrm{d}\beta_j). \tag{2.3}$$

for a (hyper)-parameter $\gamma = (\gamma_1, \ldots, \gamma_p) \in \Theta \stackrel{\text{def}}{=} [0,\infty)^p$, where for $a > 0$, p_a denotes the distribution of $\mathbf{N}(0, a)$, the Gaussian distribution on $\mathbb{R}$ with mean 0 and variance a, and $p_0(\mathrm{d}u) \stackrel{\text{def}}{=} \delta_0(\mathrm{d}u)$ denotes the Dirac measure at 0. The posterior distribution of β given $Y = y$ and given the hyper-parameter (γ, σ^2) is therefore

$$\pi_n(\mathrm{d}\beta | y, \sigma^2, \gamma) \propto \left(\frac{1}{2\pi\sigma^2}\right)^{n/2} \exp\left(-\frac{1}{2\sigma^2}\|y - X\beta\|^2\right) \pi_\gamma(\mathrm{d}\beta). \tag{2.4}$$

Sampling from the posterior distribution $\pi_n(\cdot | y, \sigma^2, \gamma)$ is straightforward. Indeed, for $\gamma = (\gamma_1, \ldots, \gamma_p) \in \Theta$, denote $\mathsf{I}_\gamma \stackrel{\text{def}}{=} \{1 \le j \le p : \ \gamma_j \neq 0\}$ the sparsity structure defined by γ. Notice that for $j \notin \mathsf{I}_\gamma$ (that is $\gamma_j = 0$), π_γ puts probability mass 1 on the event $\{\beta_j = 0\}$, and

so does the posterior distribution $\pi_n(\cdot|y,\sigma^2,\gamma)$. Hence $\pi_n(\cdot|y,\sigma^2,\gamma)$ is the distribution of the random variable $(B_1,\ldots,B_p)$ obtained by simulating $\{B_j, j\in \mathsf{I}_\gamma\}$ from $\mathbf{N}(\mu_\gamma,\sigma^2 V_\gamma)$, and by setting the remaining components to 0, where

$$\mu_\gamma = V_\gamma X_\gamma' y,\;\; V_\gamma = \left(X_\gamma' X_\gamma + \sigma^2 \bar{\Gamma}_\gamma^{-1}\right)^{-1}, \tag{2.5}$$

where X_γ is the matrix obtained from X by removing the columns j for which $\gamma_j = 0$, and $\bar{\Gamma}_\gamma$ is the diagonal matrix with diagonal elements given by $\{\gamma_j,\, j\in \mathsf{I}_\gamma\}$. With this Gaussian linear model, and prior (2.3), it is easy to check that the marginal distribution of y is $\mathbf{N}(0, C_\gamma)$, where

$$C_\gamma \stackrel{\text{def}}{=} \sigma^2 \mathbb{I}_n + \sum_{j\in \mathsf{I}_\gamma} \gamma_j x_j x_j',$$

and x_j is the j-th column of X. Therefore, up to a normalizing constant that we ignore, the log-likelihood of (σ^2,γ) is given by

$$\ell(\sigma^2,\gamma) \stackrel{\text{def}}{=} -\frac{1}{2}\log\det(C_\gamma) - \frac{1}{2}\mathsf{Tr}\left(C_\gamma^{-1} yy'\right).$$

The SBL estimator of $\beta_\star$ as proposed by [37, 88] is the empirical Bayes estimator of β given by

$$\hat{\beta}_n = \int \beta \pi_n(\mathrm{d}\beta|y,\hat{\sigma}_n^2,\hat{\gamma}_n), \tag{2.6}$$

where

$$(\hat{\sigma}_n^2,\hat{\gamma}_n) = \mathsf{Argmax}_{(\sigma^2,\gamma)\in\mathbb{R}_+\times\Theta}\, \ell(\sigma^2,\gamma). \tag{2.7}$$

Notice that $\hat{\beta}_n$ is straightforward to compute once $\hat{\sigma}_n^2$ and $\hat{\gamma}_n$ are available. Indeed given $\hat{\sigma}_n^2$ and $\hat{\gamma}_n$, $\hat{\beta}_{n,j} = 0$ for all j such that $\hat{\gamma}_{n,j} = 0$, and for the other components $j\in \mathsf{I}_{\hat{\gamma}_n}$, we have from (2.5) that

$$(\hat{\beta}_{n,j})_{j\in \mathsf{I}_{\hat{\gamma}_n}} = \left(X_{\hat{\gamma}_n}' X_{\hat{\gamma}_n} + \hat{\sigma}_n^2 \bar{\Gamma}_{\hat{\gamma}_n}^{-1}\right)^{-1} X_{\hat{\gamma}_n}' y.$$

Remark 1. *The presentation of the SBL given above is slightly different from the original presentation of [37, 88]. The key difference here is that in the prior distribution π_γ we allow the components of γ to take the value zero. This is needed for the estimator $\hat{\gamma}_n$ to be well-defined, and for the well-posedness of the question of whether the procedure produces sparse solutions.*

□

Remark 2. *The solution for β as outlined above is of the same form as the solution for ridge regression with one notable difference. In the SBL formulation, the penalization level/hyperparameters for each variable is set to the MLE while in the ridge regression*

setting, the optimal penalization level is typically set using cross-validation. In addition, if we assume H1-2 hold and $(\beta_\star)_j \neq 0$*, the estimated penalization level* $\hat{\gamma}_j \sim \mathcal{O}([\beta_\star]_j^2)$ *scales with the square magnitude of the signal/effect size. This implies that the shrinkage effect is small for large positive or negative signals. In fact, the SBL is equivalent to ridge regression with adaptive penalization.*

□

Computationally, the optimization problem (2.7) is not a "nice" problem because the objective function $\ell(\sigma^2, \gamma)$ is non-concave and typically attains its maximum at the boundary of the domain Θ (that is some of the components of its solution(s) are exactly zeros). We return to the issue of solving (2.7) in Section 2.2.3. But statistically (2.7) is interesting as it yields a sparse solution $\hat{\gamma}_n$ as we shall see.

2.2.1 Existence of $\hat{\gamma}_n$

Since the log-likelihood function ℓ is not concave in general, it is not immediately clear that the optimization problem (2.7) has a solution. The following result is a re-statement of the analysis of [37] and shows that a solution to (2.7) always exists, and is sparse. We give a straightforward proof below.

Proposition 1. *Fix* $y \in \mathbb{R}^n$*,* $X \in \mathbb{R}^{n \times p}$*, and* $\sigma^2 = \sigma_\star^2$*. Then the maximization problem* $\mathsf{Argmax}_{\gamma \in \Theta} \ell(\gamma, \sigma^2)$ *has at least one solution* $\hat{\gamma} = (\hat{\gamma}_1, \ldots, \hat{\gamma}_p)$ *which has the following property:*

$$\hat{\gamma}_j = \begin{cases} \dfrac{\left(x_j' C_j^{-1} y\right)^2 - x_j' C_j^{-1} x_j}{\left(x_j' C_j^{-1} x_j\right)^2} & \text{if } \left(x_j' C_j^{-1} y\right)^2 > x_j' C_j^{-1} x_j \\ 0 & \text{otherwise}, \end{cases} \tag{2.8}$$

where C_j *is given by*

$$C_j \overset{def}{=} \sigma^2 \mathbb{I}_n + \sum_{k \in I_{\hat{\gamma}} \setminus \{j\}} \hat{\gamma}_k x_k x_k'.$$

Proof. See Section 2.4.1. □

It is important to notice that there is no randomness involved in the above result: y and X are given and fixed. In particular we do not assume (2.1) nor (2.2). It is clear that this result does not give the expression of the maximizer since the right-hand side of (2.8) also depends on $\hat{\gamma}$. Rather it gives coherence relationships between components of the solution. But more importantly the proposition shows that the optimization problem (2.7) leads to sparse solutions $\hat{\gamma}$. One can interpret the term $x_j' C_j^{-1} y$ as a measure of correlation between

the y and the j-th column x_j of X. Hence the result shows that if the correlation between x_j and y is sufficiently weak then $\hat{\gamma}_{n,j}$ (and hence $\hat{\beta}_{n,j}$) is set exactly equal to zero. It is known from [96] Theorem 2 that even local maximum of ℓ are sparse. Proposition 1 is more precise than [96] Theorem 2, and give some insight into the sparsity structure of the global maximizer(s) of ℓ.

Of course Proposition 1 is useful only to the extent that the inequality $\left(x_j' C_j^{-1} y\right)^2 \leq x_j' C_j^{-1} x_j$ is satisfied with high probability when $\beta_{\star,j} = 0$. We investigate this below. Unfortunately we will see that in general $\hat{\gamma}_j \neq 0$ even when $(\beta_\star)_j = 0$, under the most favorable setting. We make the following distributional assumption.

H 1. *The data generating model (2.1-2.2) holds and $\epsilon \sim \boldsymbol{N}(0, \sigma_\star^2 I_n)$, for some $\sigma_\star^2 > 0$.*

We shall also focus our analysis on the idealized case where the matrix X has orthogonal columns.

H 2. *The design matrix $X \in \mathbb{R}^{n \times p}$ is such that $\langle x_k, x_j \rangle = 0$ whenever $j \neq k$ and $\langle x_k, x_k \rangle = n$.*

Proposition 2. *Suppose that H1-2 hold, and $\sigma^2 = \sigma_\star^2$. Then for any $j \in \{1, \ldots, p\}$ such that $\beta_{\star,j} = 0$,*

$$\mathbb{P}\left[\hat{\gamma}_{n,j} = 0\right] = \mathbb{P}\left[Z^2 \leq 1\right] \approx 0.68,$$

where $Z \sim \boldsymbol{N}(0,1)$.

Proof. See Section 2.4.2. □

Remark 3. *Proposition 2 holds regardless of $\sigma_\star > 0$ and n. The result might seem surprising a first. To convince the skeptical reader, consider for instance the trivial case where $n = p = 1$, and $X = 1$. In that case without even using (2.8), it is straightforward to see that the log-likelihood is $\ell(\sigma_\star, \gamma) = -(1/2)\log(\sigma_\star^2 + \gamma) - (1/2)y^2/(\sigma_\star^2 + \gamma) + \mathsf{cst}$. It is then easy to see that the maximizer of ℓ is $\hat{\gamma} = 0$ if $y^2 \leq \sigma^2$, and $\hat{\gamma} = (y^2 - \sigma_\star^2)$ is $y^2 > \sigma_\star^2$. Now, if $Y \sim \boldsymbol{N}(0, \sigma_\star^2)$, and we compute $\hat{\gamma}$ using Y, then $\hat{\gamma} = 0$ means that $Y^2 \leq \sigma_\star^2$, and this holds with probability ≈ 0.68, regardless of $\sigma_\star^2 > 0$.*

The result above shows that even in the idealized setting of H2, and under the Gaussian linear model assumption, the SBL procedure will set $\hat{\gamma}_j$ to 0 (for $j \notin \mathsf{l}$) only about 70% of the time, regardless of $\sigma_\star^2$ and the sample size. We do not know whether this result continue to hold for more general design matrices. The behavior of the solution of (2.7) for a general design matrix is technically more challenging.

Another important limitation of the SBL procedure is the computation of $\hat{\gamma}_n$ and $\hat{\sigma}_n^2$. Typically iterative methods (such as the EM algorithm, see Section 2.2.3) are used. The

EM algorithm does not promote sparsity, and converges to the solution only at the limit. Therefore, in finite time, the solutions generated by the EM algorithm are typically not sparse at all.

These two shortcomings limit the usefulness of the basic SBL procedure as an interesting method for sparse signal recovery. However, we observe that when $\beta_{\star,j} = 0$, and the condition $\left(x_j' C_{j,\hat{\gamma}_n}^{-1} Y\right)^2 \leq x_j' C_{j,\hat{\gamma}_n}^{-1} x_j$ fails, assuming again the most favorable setting of H2, $\hat{\gamma}_j$ is given

$$\hat{\gamma}_{n,j} = \frac{\sigma^2(Z_j^2 - 1)}{\langle x_j, x_j \rangle},$$

where $Z_j \sim \mathbf{N}(0,1)$. Hence $\hat{\gamma}_j$ has mean zero and variance of order $O(\|x_j\|^{-4}) \approx O(n^{-2})$. We conclude that when SBL fails to set to zero a component j such that $\beta_{\star,j} = 0$, the computed SBL solution $\hat{\gamma}_j$ is typically very small. This suggests that a thresholded version of $\hat{\gamma}_n$ should be able to set these terms to zero. We pursue this approach in Section 2.2.2.

2.2.2 A thresholded version and its statistical properties

We saw in Section 2.2.1 that although sparse, $\hat{\gamma}_n$ does not recover in general the sparsity structure of $\beta_\star$. To improve on this we propose a modified, hard-thresholded version of $\hat{\gamma}_n$ denoted $\tilde{\gamma}_n$ and defined as follows. For $1 \leq j \leq p$,

$$\tilde{\gamma}_{n,j} \stackrel{\text{def}}{=} \begin{cases} \hat{\gamma}_{n,j} & \text{if } \hat{\gamma}_{n,j} > \frac{\hat{\sigma}_n^2 z_\star}{\|x_j\|^2} \\ 0 & \text{otherwise} \end{cases}, \tag{2.9}$$

for a thresholding parameter $z_\star$. The corresponding modified estimator of $\beta_\star$ is

$$\tilde{\beta}_n \stackrel{\text{def}}{=} \int \beta \pi_n(\mathrm{d}\beta | y, \hat{\sigma}_n^2, \tilde{\gamma}_n).$$

Theorem 1. *Assume H1-2, and suppose that $\sigma_\star^2$ is known,* $\log s \geq 1$*, and $z_\star = c_0 \log p$, for some constant $c_0 > 2$, where $s = |I_{\gamma_\star}|$. Then*

$$\|\tilde{\beta}_n - \beta_\star\|_2 \leq 2\sigma\sqrt{2 + c_0}\sqrt{\frac{s \log(p)}{n}}, \tag{2.10}$$

with probability at least $1 - \frac{1}{p^{(c_0 s)/8}} - \frac{1}{\exp(s)}$.

Proof. See Section 2.4.3. □

We deduce the following corollary. For $u \in \mathbb{R}^p$, $\mathsf{sign}(u) = (s_1, \ldots, s_p)$ where for each i, $s_i = 0$ if $u_i = 0$, $s_i = 1$ is $u_i > 0$, and $s_i = -1$ if $u_i < 0$.

Corollary 1. *In addition to the assumptions of Theorem 1, suppose that*

$$\min_{\{j:\,|\beta_{\star,j}|>0\}} |\beta_{\star,j}| > 2\sigma\sqrt{2+c_0}\sqrt{\frac{s\log(p)}{n}}. \tag{2.11}$$

Then with probability at least $1 - \frac{1}{p^{\frac{c_0 s}{8}}} - \frac{1}{\exp(s)} - \frac{1}{p^{\frac{c_0}{2}-1}}$, $\mathsf{sign}(\tilde{\beta}_n) = \mathsf{sign}(\beta_\star)$.

Proof. See Section 2.4.4. □

Remark 4. *Theorem 1 shows that the thresholded SBL achieves the same ℓ^2 convergence rate as LASSO (see for instance **Corollary 2** in [68] and the references therein). And Corollary 1 shows that the estimator recovers the sparsity structure of the signal with high-probability. Theorem 1 is obtained under the orthonormal design matrix assumption (H2) that is much stronger than the restricted eigenvalue assumption typically needed for LASSO ([13, 68]). However, the simulation results reported below suggest that these results continue to hold more broadly.*

Remark 5. *Theorem 1 suggests using $z_\star = c_0 \log(p)$. As a practical matter, we need a value of c_0 to implement the thresholded SBL. We propose setting $c_0 = c(1 + |\hat{\rho}|)$, for a constant c, and where $\hat{\rho}$ is an estimate of the largest correlation among the columns of X. We found that $c \approx 2$ gives reasonably good value. Alternatively, one can tune c using BIC.*

2.2.3 Computing $\hat{\sigma}_n^2$ and $\hat{\gamma}_n$

Here we address the issue of solving (2.7). Because the function $\ell(\sigma^2, \gamma)$ is not concave, and typically attains its maximum at the boundary of the domain Θ, the optimization (2.7) is not a smooth problem. The strategy originally developed by [88] focuses instead on the smooth problem obtained by maximizing ℓ over the open domain $\mathbb{R}+^{p+1}$, where $\mathbb{R}+ \stackrel{\text{def}}{=} (0, \infty)$. That is, find

$$\mathsf{Argmax}_{(\sigma^2,\gamma)\in\mathbb{R}_+^{p+1}}\ \ell(\sigma^2, \gamma). \tag{2.12}$$

The EM algorithm proposed by [88] for solving (2.12) is as follows.

2.2.3.1 EM algortihm

repeat

Given $(\{\sigma^2\}^{(k)}, \gamma^{(k)}) \in \mathbb{R}_+^{p+1} = (0, \infty)^{p+1}$, we compose the matrix $\Gamma^{(k)} = \mathsf{diag}(\gamma_1^{(k)}, \ldots, \gamma_p^{(k)})$.

Compute $V^{(k)} = \left(X'X + \{\sigma^2\}^{(k)}\{\Gamma^{(k)}\}^{-1}\right)^{-1}$, and $\mu^{(k)} = V^{(k)}X'y$.

Set

$$\gamma_j^{(k+1)} = \{\mu_j^{(k)}\}^2 + \{\sigma^2\}^{(k)} V_{j,j}^{(k)}, \quad j = 1, \ldots, p,$$
$$\{\sigma^2\}^{(k+1)} = \frac{1}{n}\left(\|y - X\mu^{(k)}\|^2 + \{\sigma^2\}^{(k)} \mathsf{Tr}(V^{(k)} X'X)\right).$$

until convergence

Clearly, Problem (2.12) has no solution whenever the solution of (2.7) occurs at the boundary of Θ. Nevertheless, we will see that the above EM algorithm produces sequences that converge to the solution of (2.7). To simplify the analysis we assume again that H2 holds and that σ^2 is fixed. Hence we focus only on the recursion in γ:

$$\gamma_j^{(k+1)} = \{\mu_j^{(k)}\}^2 + \sigma^2 V_{j,j}^{(k)}, \quad j = 1, \ldots, p.$$

With the assumption that the design matrix is orthogonal, we can work out explicitly the terms $V^{(k)} = \left(X'X + \sigma^2\{\Gamma^{(k)}\}^{-1}\right)^{-1}$ and $\mu^{(k)} = V^{(k)}X'y$, which leads to

$$\gamma_j^{(k+1)} = \frac{\langle x_j, y\rangle^2}{\left(\|x_j\|^2 + \frac{\sigma^2}{\gamma_j^{(k)}}\right)^2} + \frac{\sigma^2}{\|x_j\|^2 + \frac{\sigma^2}{\gamma_{j^{(k)}}}}, \quad j = 1, \ldots, p. \tag{2.13}$$

Proposition 3. *Fix $y \in \mathbb{R}^n$, and $X \in \mathbb{R}^{n\times p}$ such that H2 holds. Fix $\sigma^2 > 0$. Let $\{\gamma^{(k)}, k \geq 0\}$ denote the sequence produced by the recursion (2.13) for some initial $\gamma^{(0)}$ with positive components. Then for all $j \in \{1, \ldots, p\}$,*

$$\lim_{k\to\infty} \gamma_j^{(k)} = \hat{\gamma}_{n,j} = \begin{cases} \frac{\langle y, x_j\rangle^2 - \sigma^2 n}{n} & \text{if } \langle y, x_j\rangle^2 > \sigma^2 n \\ 0 & \text{otherwise} \end{cases}.$$

Proof. See Section 2.4.5 □

2.2.4 A simulation study

We investigate by simulation the behavior of the proposed thresholded SBL. The behavior of the method described in Theorem 1 and Corollary 1 are obtained under the very strong assumption H2, which implies in particular that $p \leq n$. The simulation study in this section explores how the method holds up in settings where some of these assumptions fail.

2.2.4.1 Synthetic Data Sets

We investigate by simulation the behavior of the SBL procedure and its thresholded version, and how they compare with LASSO. For all the simulations, $n = 100$ and $p = 500$. We generate the design matrix X by simulating each row independently from the Gaussian

distribution $\mathbf{N}(0,\Sigma)$ where $\Sigma_{ii}=1$ and $\Sigma_{ij}=\rho$ for $i\neq j$. We consider two values of ρ: $\rho=0$ for which X is close to satisfy H2, and $\rho=0.9$ which produces a design matrix X with strongly correlated variables. We simulate the dependent variable Y from the $\mathbf{N}(X\beta_\star,\sigma_\star^2\mathbb{I}_n)$, with $\sigma_\star=1$. We consider four (4) different scenarios of sparsity, with $s=3,15,25$, and $s=50$ where s is the number of non-zero elements of $\beta_\star$. The magnitude of the non-zero elements also play an important role in the recovery. We generate all the non-zeros components of $\beta_\star$ from the uniform distribution $\mathbf{U}(a,a+1)$, for a ranging from 0 to 9. In addition, the sign of the non-zeros are determined via a coin flip.

For each value of ρ, each sparsity level, and each signal strength a, we repeat each estimator 30 times, and we compute the relative error rate ($\|\hat{\beta}-\beta_\star\|/\|\beta_\star\|$), the sensitivity and the specificity, averaged over these 30 replications. The sensitivity (SEN) and the specificity (SPE) of a given estimator $\hat{\beta}$ are defined as

$$\mathsf{SEN}(\hat{\beta})=\frac{\sum_{j=1}^{p}\mathbf{1}_{\{\hat{\beta}_j\neq 0\}}\mathbf{1}_{\{\beta_{\star,j}\neq 0\}}}{\sum_{j=1}^{p}\mathbf{1}_{\{\beta_{\star,j}\neq 0\}}},\quad\text{and}\quad \mathsf{SPE}(\hat{\beta})=\frac{\sum_{j=1}^{p}\mathbf{1}_{\{\hat{\beta}_j\neq 0\}}\mathbf{1}_{\{\beta_{\star,j}\neq 0\}}}{\sum_{j=1}^{p}\mathbf{1}_{\{\hat{\beta}_j\neq 0\}}}.$$

These measures are valid for any estimator $\hat{\beta}$, and we compute them for the thresholded version of SBL, the non-thresholded version of SBL, as well as for the LASSO estimator. For the thresholded SBL, we use $z_\star=c(1+|\hat{\rho}|)\log p$, where c is determined by minimizing the BIC: $\frac{\|y-X\hat{\beta}\|}{2\hat{\sigma}^2}+s\log(n)$.

We compute the LASSO estimator using the function *cv.glmnet* of the package GLMNet ([40]) where we select the penalty term λ by a 10-fold cross-validation procedure. In the cross-validation, the regulation parameter selected minimizes the prediction error.

The simulation results are presented in Figure 2.1-2.8. As one can see from these figures, the main conclusion is that SBL is more sensitive than LASSO to the strength of the signal (defined here as as the parameter a). With a weak signal it performs poorly, but outperforms LASSO when the signal is strong enough. Another interesting finding is that, overall, LASSO performs poorly in selecting the non-zeros components (variable selection). This is consistent with recent results ([60]) which shows that variable selection consistency of LASSO requires the irrepresentable condition, which actually is a very strong condition that often does not hold in practice. For instance, the irrepresentable condition fails for all the design matrices of this simulation study, except for the design matrix behind Figure 2.2.

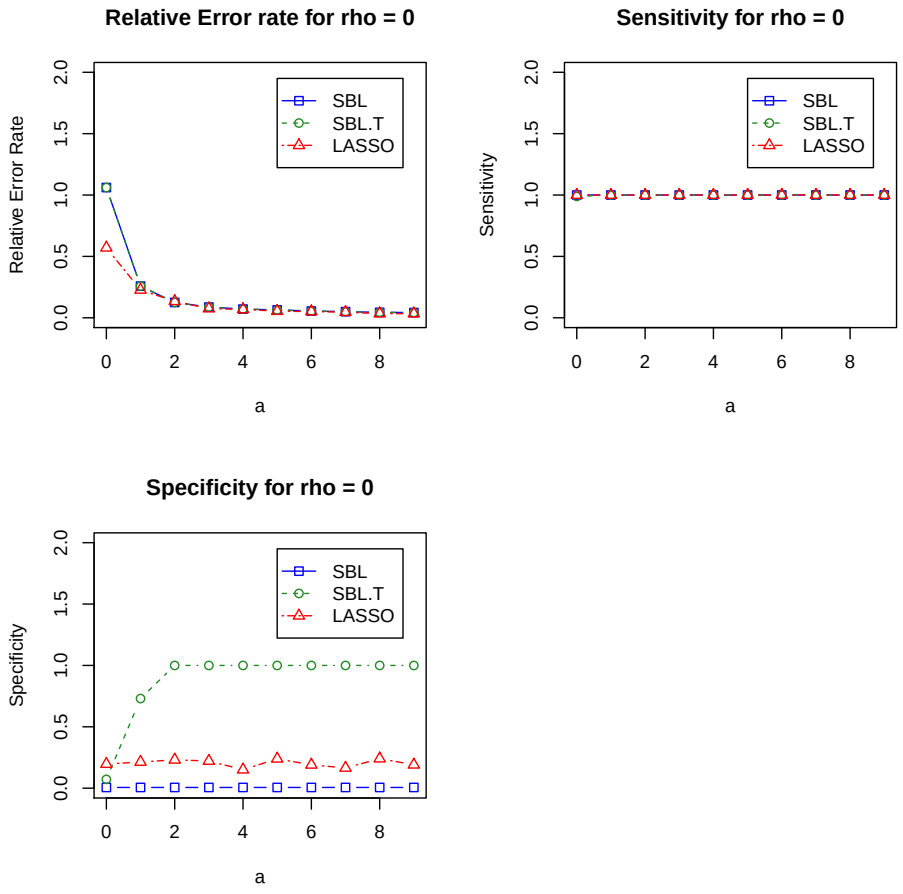


Figure 2.1: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 3$.

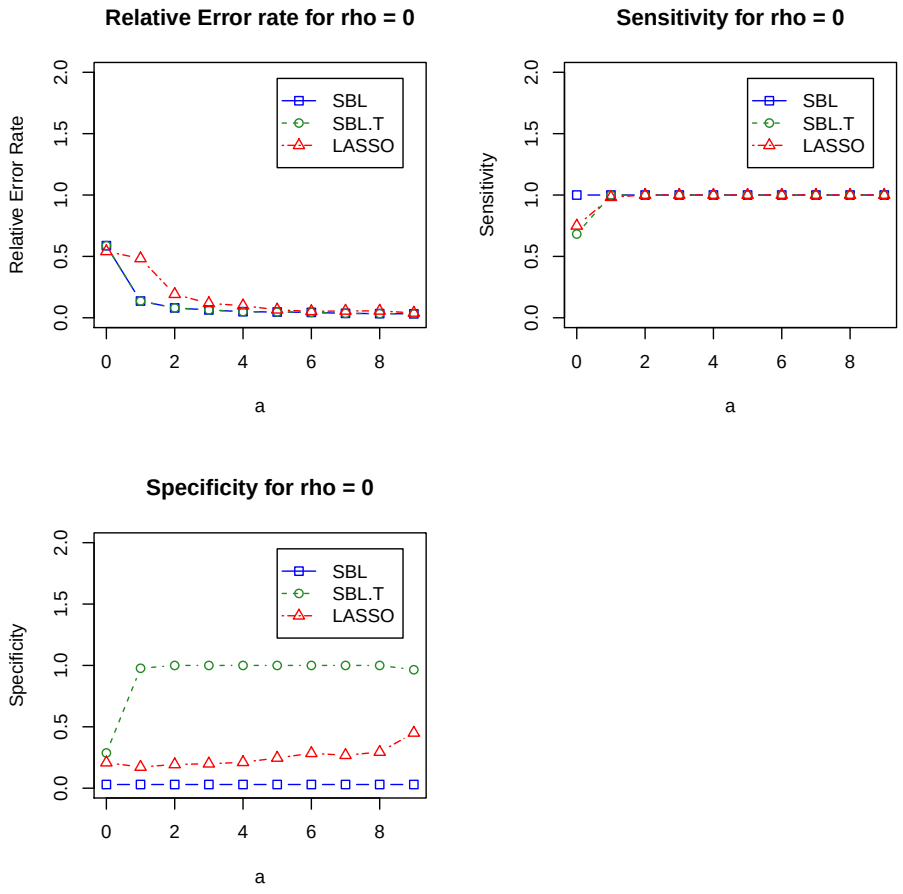


Figure 2.2: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 15$.

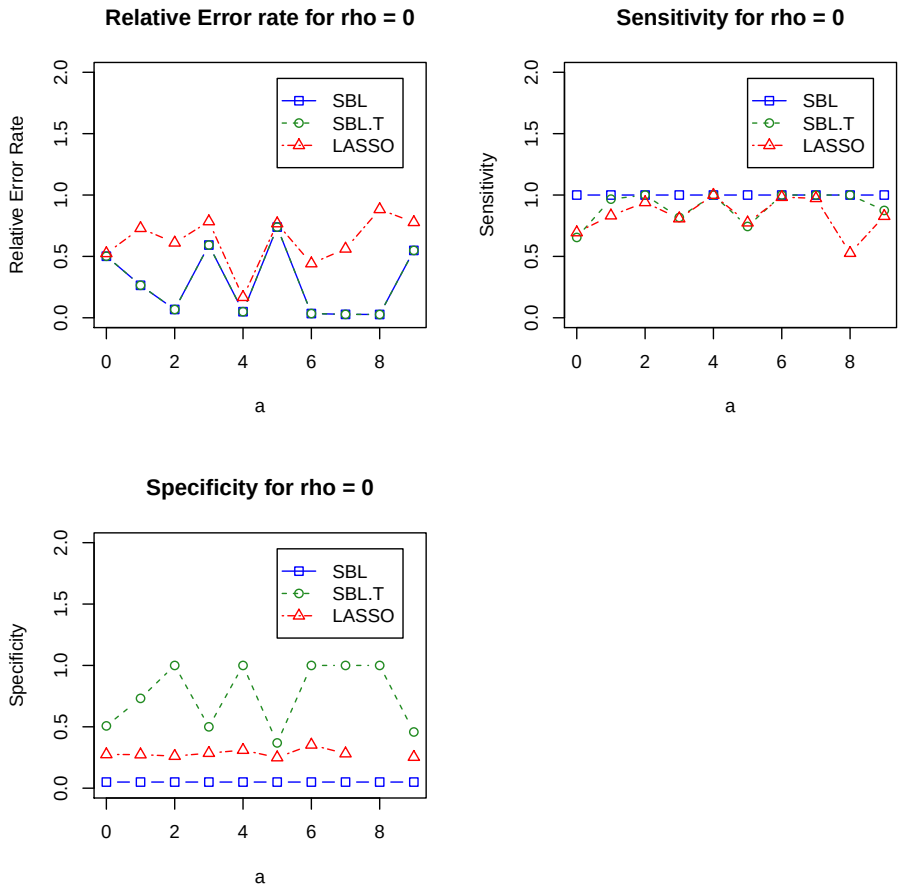


Figure 2.3: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 25$.

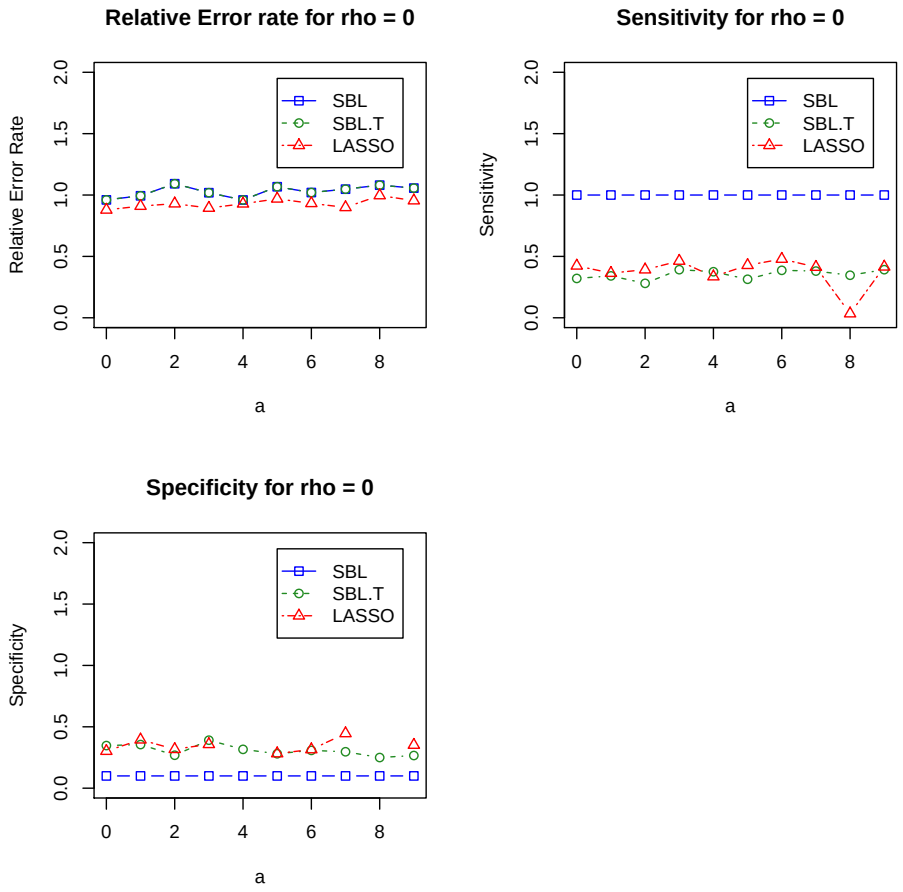


Figure 2.4: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 50$.

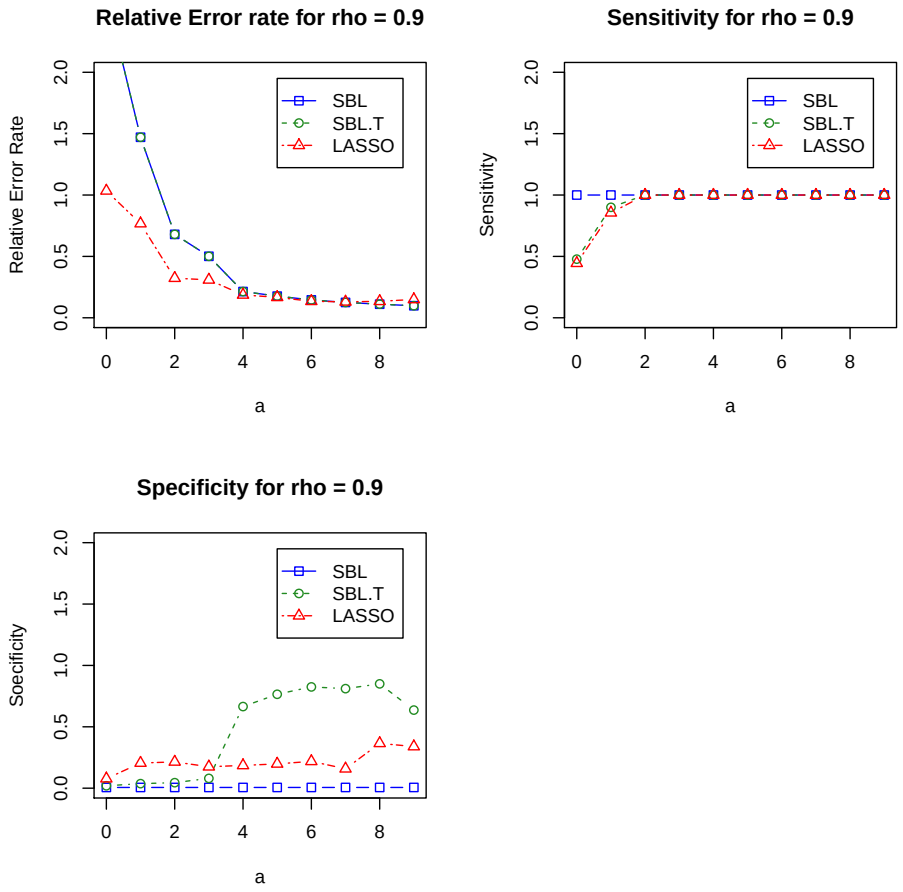


Figure 2.5: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 3$.

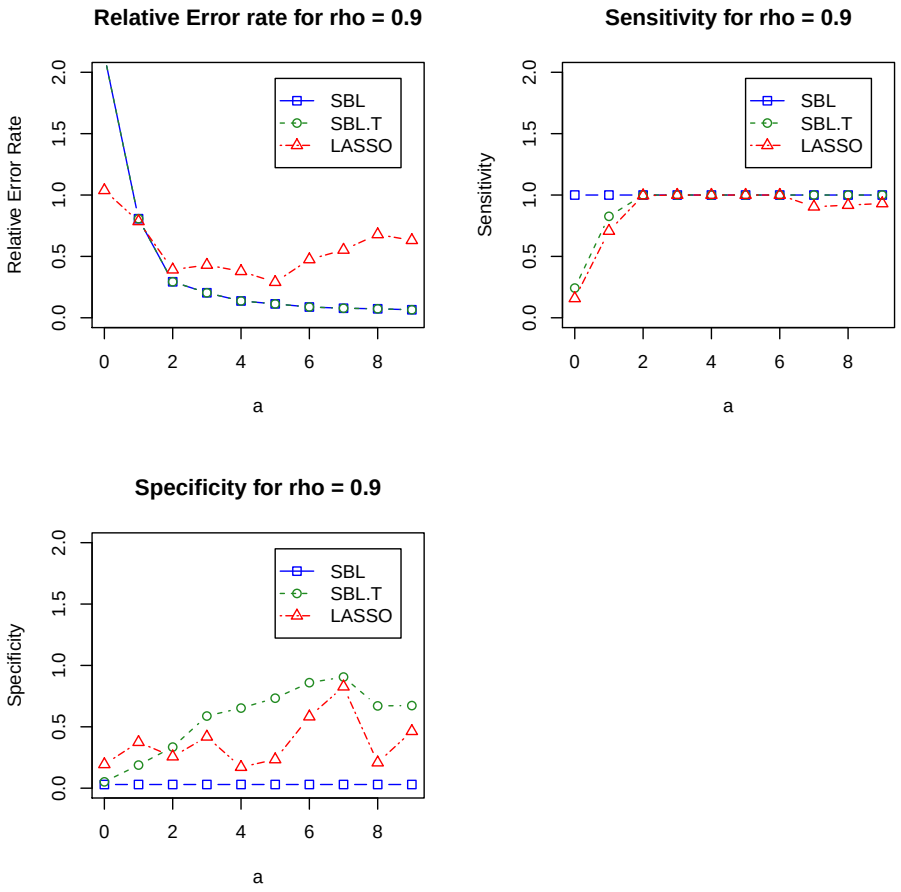


Figure 2.6: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 15$.

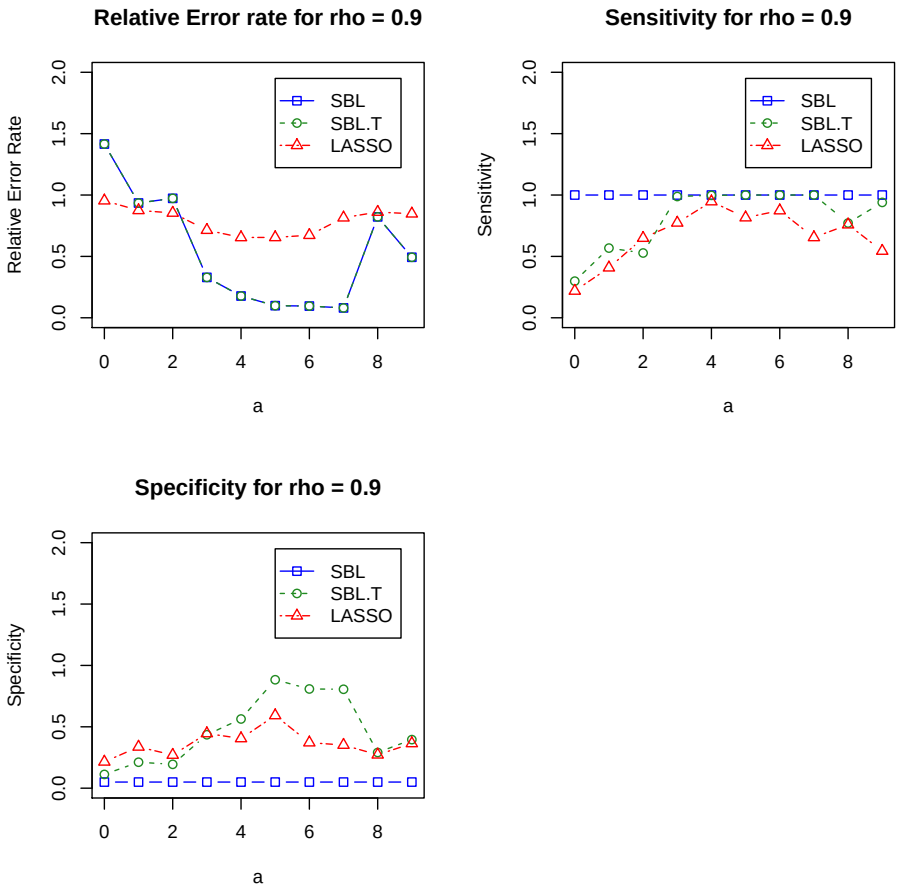


Figure 2.7: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 25$.

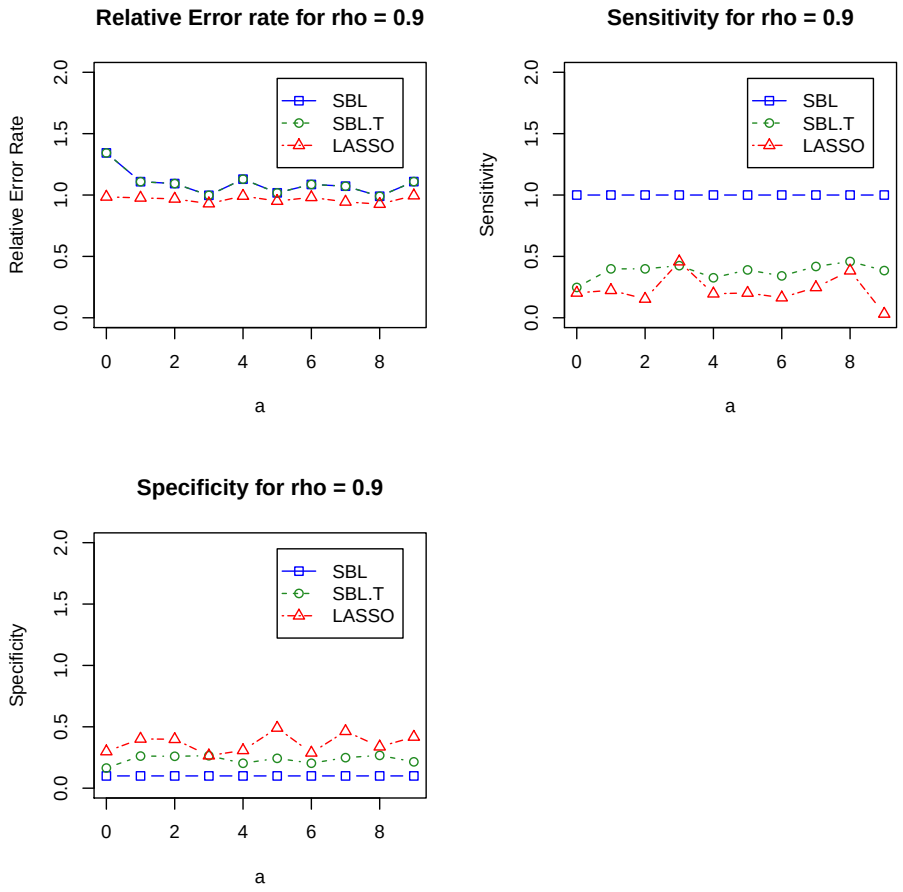


Figure 2.8: Sensitivity, specificity and relative error for SBL and LASSO as function of a. $s = 50$.

2.2.4.2 A simulated real data example

In this example, we consider a micro-array data concerning genes involved in the production of riboflavin. The data is made publicly available at

`http://www.annualreviews.org/doi/suppl`
`/10.1146/annurev-statistics-022513-115545`

and contains $n = 71$ samples and $p = 4088$ covariates corresponding to 4088 genes. Each of the sample contains a real valued response consisting of the logarithm of the riboflavin production rate and 4088 real valued covariates consisting of the logarithm of the genes' expression levels.

Given the very high dimensionality of this dataset, the lack of any true value of the parameter, and given also the fact that micro-array data are well-known to be very noisy,

direct comparison of different regression methods on such dataset cannot be very insightful. For a more meaningful comparison, we use the riboflavin design matrix $X \in \mathbb{R}^{71\times4088}$ to generate simulated levels of riboflavin production rate using the sparse regression model $Y = X\beta + \epsilon$ where $\epsilon \sim N(0, \sigma^2\mathbb{I}_{71})$, with $\sigma^2 = 1$. The magnitude of the non-zero components of β are uniformly simulated $\beta_j \sim U(a, a+1)$ with $a = \{0, 1, \ldots, 9\}$. We set the number of non-zeros elements in the vector β to 5. Figure 2.9 shows the results of the simulation evaluated using the aforementioned metrics. Under such extreme high-dimensional conditions, both methods perform poorly. SBL has found all the relevant variables but has also selected many non-relevant variables. LASSO has produced more sparse solutions, but has missed some important variables. The results remain essentially the same even when we set σ^2 (the variance of the noise ϵ) to 0.1.

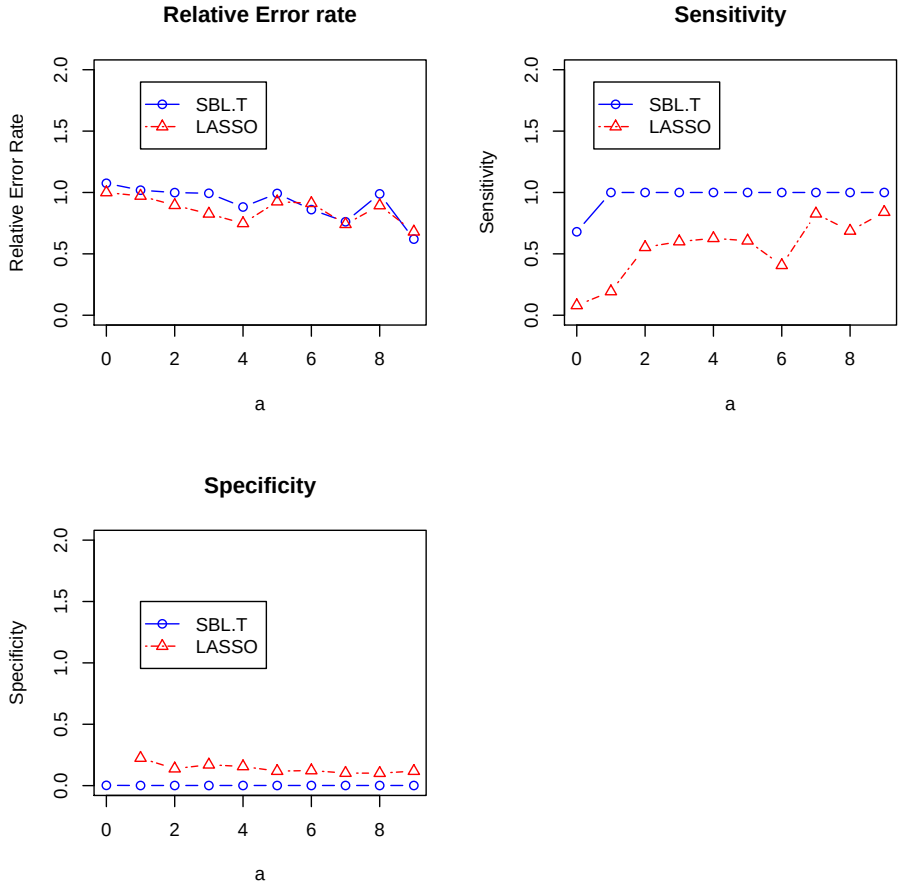


Figure 2.9: Sensitivity, specificity and relative error for SBL(thresholded) and LASSO as function of a. $s = 5$. The results of this simulation is generated using the riboflavin data

One final word on computing times. We compute the SBL estimate using Algorithm 2.2.3.1, and we use the package GLMNet to compute LASSO. We implemented Algorithm 2.2.3.1 in R. The core of the GLMNet package is written in Fortran and the result is very fast. The comparison of the computing times is largely in favor of GLMNet. Comparing computing times is always tricky as it depends to a large extent on the programming language and skills. But beyond the implementation differences, it seems clear that LASSO has a computational advantage over SBL in that it leads to "easier" (convex) optimization problems, compared to SBL.

2.3 Conclusion

The contribution of this chapter is two-fold. First, we show that in high-dimensional regression with sparse signals, the plain SBL estimator does not recover the sparsity structure of the signal in general. To remedy this, we have proposed a hard-thresholded version of the SBL estimator. We show that for orthogonal design matrices, this hard-thresholded version of the SBL achieves the same ℓ^2 convergence rate of $M\sigma\sqrt{s\log p/n}$, as LASSO. We have also established that with high-probability the thresholded estimator recovers the sparsity structure of the signal. Furthermore our simulation results show that the thresholded estimator compares very well with LASSO, and outperforms LASSO when the signal is strong.

One important and pressing issue is the extension of these results to non-orthogonal design matrices. In particular we wish to understand the type of design matrix X for which these results continue to hold. This SBL theory and its comparison with the recently developed LASSO theory (see for instance [13, 60]) could potentially give new insight into high-dimensional regression analysis. The generalized singular value decomposition (see e.g. [43]) of X_γ and Γ_γ seems to be a promising approach to tackle this problem. The challenge in this approach appears to be the development of an appropriate differentiability theory for the components of the GSVD decomposition as a function of γ.

The SBL method can be extended in several directions. It can be easily extended to deal with generalized linear models, and graphical models. But in these extensions, the computation of the estimator might require some new algorithms. Another possible extension of the method would be to replace the Gaussian distribution in the prior π_γ by some other distribution. Some work in this direction include [10, 97].

2.4 Proofs

2.4.1 Proof of Proposition 1

Proof. For any $i \in \{1, \ldots, p\}$,

$$\ell(\gamma) \leq -\frac{1}{2}\log\det(C_\gamma) \leq -\frac{1}{2}\log\det\left(\sigma^2\mathbb{I}_n + \gamma_i x_i x_i'\right) \downarrow -\infty,$$

as $\gamma_i \to \infty$. This together with the continuity of $\gamma \mapsto \ell(\gamma)$ imply the existence of a maximizer. For any such maximizer $\hat{\gamma}$, consider the vector γ such that the j-th component of γ is free to vary and the remaining components γ_{-j} are fixed to $\hat{\gamma}_{-j}$. Then we write $C_\gamma = C_j + \gamma_j x_j x_j'$ and use the matrix identity $(A + uu')^{-1} = A^{-1} - \frac{A^{-1}uu'A^{-1}}{1+u'A^{-1}u}$ to deduce that

$$C_\gamma^{-1} = C_j^{-1} - \frac{\gamma_j C_j^{-1} x_j x_j' C_j^{-1}}{1 + \gamma_j x_j' C_j^{-1} x_j}.$$

Therefore,

$$\ell(\gamma) = -\frac{1}{2}\log\det\left(C_j + \gamma_j x_j x_j'\right) + \frac{1}{2}\frac{\gamma_j\left(x_j' C_j^{-1} y\right)^2}{1 + \gamma_j x_j' C_j^{-1} x_j} - \frac{1}{2} y' C_j^{-1} y.$$

Since C_j does not depend on γ_j, we easily see that $\gamma_j \mapsto \ell(\gamma)$ is differentiable on $(0, \infty)$ and

$$\begin{aligned}
\frac{\partial}{\partial\gamma_j}\ell(\gamma) &= -\frac{1}{2}x_j' C_j^{-1} x_j + \frac{1}{2}\frac{\gamma_j\left(x_j' C_j^{-1} x_j\right)^2}{1 + \gamma_j x_j' C_j^{-1} x_j} + \frac{1}{2}\frac{\left(x_j' C_j^{-1} y\right)^2}{\left(1 + \gamma_j x_j' C_j^{-1} x_j\right)^2} \\
&= \frac{1}{2}\frac{\left(x_j' C_j^{-1} y\right)^2 - \gamma_j\left(x_j' C_j^{-1} x_j\right)^2 - x_j' C_j^{-1} x_j}{\left(1 + \gamma_j x_j' C_j^{-1} x_j\right)^2}
\end{aligned}$$

If $\left(x_j' C_j^{-1} y\right)^2 \leq x_j' C_j^{-1} x_j$, $\frac{\partial}{\partial\gamma_j}\ell(\gamma) < 0$ and $\gamma_j \mapsto \ell(\gamma)$ attains its maximum at 0. Similarly if $\left(x_j' C_j^{-1} y\right)^2 > x_j' C_j^{-1} x_j$, it is easy to check that $\gamma_j \mapsto \ell(\gamma)$ attains its maximum at $(\left(x_j' C_j^{-1} y\right)^2 - x_j' C_j^{-1} x_j)/\left(x_j' C_j^{-1} x_j\right)^2$. Now if $\hat{\gamma}_j$ differs from the maximizer just found, we can improve on the likelihood by setting $\hat{\gamma}_j$ equal to that maximizer, which would be a contradiction. Hence the result. □

2.4.2 Proof of Proposition 2

Proof. Recall that $\mathsf{I} = \{1 \le j \le p : \beta_{\star,j} \neq 0\}$ is the sparsity structure of $\beta_\star$. For $\gamma \in \Theta$, and $1 \le j \le p$, we define $\mathsf{I}_0 \stackrel{\text{def}}{=} \mathsf{I} \cap \mathsf{I}_\gamma \setminus \{j\}$, and $\mathsf{I}_1 \stackrel{\text{def}}{=} \mathsf{I}^c \cap \mathsf{I}_\gamma \setminus \{j\}$, where in order to keep the notation easy, we omit the dependence of I_0 and I_1 on (γ, j). We will also write X_{I_0} (resp. X_{I_1}) to denote the matrix obtained by collecting the columns of X whose indexes belong to I_0 (resp. I_1). We define

$$\begin{aligned} C_{j,\gamma} &\stackrel{\text{def}}{=} \sigma^2 \mathbb{I}_n + \sum_{k \in \mathsf{I}_\gamma \setminus \{j\}} \gamma_k x_k x_k' \\ &= \sigma^2 \mathbb{I}_n + \sum_{k \in \mathsf{I}_0} \gamma_k x_k x_k' + \sum_{k \in \mathsf{I}_1} \gamma_k x_k x_k'. \end{aligned}$$

By the Woodbury matrix identity and the assumption $X_{\mathsf{I}_0}' X_{\mathsf{I}_1} = 0$, we get:

$$C_{j,\gamma}^{-1} = \frac{1}{\sigma^2}\mathbb{I}_n - \frac{1}{\sigma^4} X_{\mathsf{I}_0} \left(\Gamma_{\mathsf{I}_0}^{-1} + \frac{1}{\sigma^2} X_{\mathsf{I}_0}' X_{\mathsf{I}_0} \right)^{-1} X_{\mathsf{I}_0}' - \frac{1}{\sigma^4} X_{\mathsf{I}_1} \left(\Gamma_{\mathsf{I}_1}^{-1} + \frac{1}{\sigma^2} X_{\mathsf{I}_1}' X_{\mathsf{I}_1} \right)^{-1} X_{\mathsf{I}_1}'.$$

Hence, for $k \in \mathsf{I}$, and using the fact that $j \notin \mathsf{I}$, we have

$$x_j' C_{j,\gamma}^{-1} x_k = 0, \text{ and } x_j' C_{j,\gamma}^{-1} x_j = \frac{1}{\sigma^2} n.$$

Therefore, if $Y = X\beta_\star + \epsilon$, we get

$$x_j' C_{j,\gamma}^{-1} Y = \frac{1}{\sigma^2} \langle x_j, \epsilon \rangle \sim \mathbf{N}\left(0, \sigma^2 n\right).$$

Now, the matrix C_j defined in Proposition 1 is $C_j = C_{j,\hat{\gamma}_n}$. Hence

$$\mathbb{P}\left[\hat{\gamma}_{n,j} = 0\right] = \mathbb{P}\left[\left(x_j' C_{j,\hat{\gamma}_n}^{-1} Y\right)^2 \le x_j' C_{j,\hat{\gamma}_n}^{-1} x_j\right] = \mathbb{P}\left[Z^2 \le 1\right],$$

where $Z \sim \mathbf{N}(0,1)$. Hence the result. □

2.4.3 Proof of Theorem 1

Proof. Under H2, $x_j C_{j,\hat{\gamma}_n}^{-1} Y = \langle x_j, Y \rangle / \sigma^2$, and $x_j C_{j,\hat{\gamma}_n}^{-1} x_j = n/\sigma^2$. Hence

$$\tilde{\gamma}_j = \begin{cases} \frac{\langle x_j, Y \rangle^2 - \sigma^2 n}{n} & \text{if } \langle x_j, Y \rangle^2 > \sigma^2 n (1 + z_\star) \\ 0 & \text{otherwise.} \end{cases}$$

Similarly, under H1 $\hat{\beta}_{n,j}$ has the explicit form $\hat{\beta}_{n,j} = \frac{\langle Y, X_j \rangle}{n + \frac{\sigma^2}{\tilde{\gamma}_{n,j}}}$. It follows that

$$\tilde{\beta}_{n,j} = \begin{cases} \frac{\langle Y, x_j \rangle}{n + \frac{\sigma^2}{\tilde{\gamma}_{n,j}}} & \text{if } \langle Y, x_j \rangle^2 > \sigma^2 n(1 + z_\star) \\ 0 & \text{otherwise .} \end{cases}$$

Again using the orthogonality assumption of X, we obtain $\langle Y, x_j \rangle = \beta_{\star,j} n + \langle \epsilon, X_j \rangle$. We set $t_j \stackrel{\text{def}}{=} \langle \epsilon, X_j \rangle$. Then it follows that

$$\tilde{\beta}_{n,j} - \beta_{\star,j} = \begin{cases} 0 & \text{if } \beta_{\star,j} = 0, \text{ and } \left(\frac{t_j}{n}\right)^2 \leq \frac{\sigma^2}{n}(1 + z_\star) \\ \frac{t_j}{n + \frac{\sigma^2}{\tilde{\gamma}_{n,j}}} & \text{if } \beta_{\star,j} = 0, \text{ and } \left(\frac{t_j}{n}\right)^2 > \frac{\sigma^2}{n}(1 + z_\star) \\ -\beta_{\star,j} & \text{if } \beta_{\star,j} \neq 0 \text{ and } \left(\frac{t_j}{n} + \beta_{\star,j}\right)^2 \leq \frac{\sigma^2}{n}(1 + z_\star) \\ \frac{n\beta_{\star,j} + t_j}{n + \frac{\sigma^2}{\tilde{\gamma}_{n,j}}} - \beta_{\star,j} & \text{if } \beta_{\star,j} \neq 0 \text{ and } \left(\frac{t_j}{n} + \beta_{\star,j}\right)^2 > \frac{\sigma^2}{n}(1 + z_\star). \end{cases}$$

Suppose that $j \in \mathsf{I} = \mathsf{I}_{\gamma_\star}$ and $\left(\frac{t_j}{n} + \beta_{\star,j}\right)^2 \leq \frac{\sigma^2}{n}(1 + z_\star)$. Then with $Z_j \stackrel{\text{def}}{=} \frac{t_j}{\sigma\sqrt{n}}$,

$$|\beta_{\star,j}| \leq \left|\frac{t_j}{n}\right| + \frac{\sigma}{\sqrt{n}}\sqrt{1 + z_\star} = \frac{\sigma}{\sqrt{n}}\left(|Z_j| + \sqrt{1 + z_\star}\right).$$

Hence for such index j,

$$\beta_{\star,j}^2 \leq \frac{2\sigma^2}{n}\left(Z_j^2 + 1 + z_\star\right). \tag{2.14}$$

But for $j \in \mathsf{I}_{\gamma_\star}$, such that $\left(\frac{t_j}{n} + \beta_{\star,j}\right)^2 > \frac{\sigma^2}{n}(1 + z_\star)$, $\tilde{\gamma}_{n,j} = \left(\frac{t_j}{n} + \beta_{\star,j}\right)^2 - \frac{\sigma^2}{n}$. Using this with some easy algebra, we obtain that for such index j,

$$\frac{n\beta_{\star,j} + t_j}{n + \frac{\sigma^2}{\tilde{\gamma}_{n,j}}} - \beta_{\star,j} = \frac{t_j}{n} - \frac{\sigma^2}{t_j + n\beta_{\star,j}}. \tag{2.15}$$

It follows that

$$\left|\frac{n\beta_{\star,j} + t_j}{n + \frac{\sigma^2}{\tilde{\gamma}_{n,j}}} - \beta_{\star,j}\right| \leq \frac{\sigma}{\sqrt{n}}\left(1 + |Z_j|\right). \tag{2.16}$$

With (2.14) and (2.16), we get

$$\sum_{j \in \mathsf{I}_{\gamma_\star}} \left(\tilde{\beta}_{n,j} - \beta_{\star,j}\right)^2 \leq \frac{2\sigma^2}{n} \sum_{j \in \mathsf{I}_{\gamma_\star}} (1 + z_\star + Z_j^2),$$

Set $s \stackrel{\text{def}}{=} |I_{\gamma_\star}|$. By [86] Lemma 5, $\mathbb{E}(|Z_j^2 - 1|^k) \le k! 2^{k-2}$, $k > 2$. Hence by Bernstein's inequality (see e.g. [91] Lemma 2.2.11), we conclude that

$$\mathbb{P}\left[\sum_{j \in \mathsf{I}_{\gamma_\star}} (1 + z_\star + Z_j^2) > 2(1 + z_\star)s\right] \le \mathbb{P}\left[\sum_{j \in \mathsf{I}_{\gamma_\star}} (Z_j^2 - 1) > z_\star s\right] \le \exp\left(-\frac{z_\star^2 s^2}{4(1 + z_\star s)}\right) \le \exp\left(-\frac{z_\star s}{8}\right) \le \frac{1}{p^{c_0 s/8}}.$$

Hence with probability at least $1 - \frac{1}{p^{c_0 s/8}}$,

$$\sum_{j \in \mathsf{I}_{\gamma_\star}} \left(\tilde{\beta}_{n,j} - \beta_{\star,j}\right)^2 \le \frac{4\sigma^2}{n}(1 + z_\star)\, s \le \frac{4(1 + c_0)\sigma^2 s \log p}{n}. \tag{2.17}$$

On the other hand, from (2.15), $\frac{t_j}{n + \frac{\sigma^2}{\tilde{\gamma}_{n,j}}} = \frac{t_j}{n} - \frac{\sigma^2}{t_j}$, hence

$$\sum_{j \notin \mathsf{I}_{\gamma_\star}} \left(\tilde{\beta}_{n,j} - \beta_{\star,j}\right)^2 = \sum_{j \notin \mathsf{I}_{\gamma_\star}, Z_j^2 > 1 + z_\star} \frac{\sigma^2}{n}\left(Z_j - \frac{1}{Z_j}\right)^2 \le \frac{\sigma^2}{n} \sum_{j \notin \mathsf{I}_{\gamma_\star}} Z_j^2 \mathbf{1}_{\{Z_j^2 > 1 + z_\star\}} \le \frac{\sigma^2}{n} \sum_{j=1}^{p} Z_j^2 \mathbf{1}_{\{Z_j^2 > 1 + z_\star\}}.$$

Now for any $\kappa \in (0, 1/2)$, $a > 0$, and by Markov's inequality

$$\begin{aligned} \mathbb{P}\left[\sum_{j=1}^{p} Z_j^2 \mathbf{1}_{\{Z_j^2 > 1 + z_\star\}} > a\right] &= \mathbb{P}\left[\exp\left(\sum_{j=1}^{p} \kappa Z_j^2 \mathbf{1}_{\{Z_j^2 > 1 + z_\star\}}\right) > e^{a\kappa}\right] \\ &\le \exp\left[-a\kappa + p \log \mathbb{E}\left[\exp\left(\kappa Z_1^2 \mathbf{1}_{\{Z_1^2 > 1 + z_\star\}}\right)\right]\right]. \end{aligned} \tag{2.18}$$

We calculate that

$$\begin{aligned} \mathbb{E}\left[\exp\left(\kappa Z_1^2 \mathbf{1}_{\{Z_1^2 > 1 + z_\star\}}\right)\right] &= 2\int_0^{\sqrt{1+z_\star}} \frac{e^{-x^2/2}}{\sqrt{2\pi}} \mathrm{d}x + 2\int_{\sqrt{1+z_\star}}^{\infty} \frac{e^{-\frac{1}{2}(1-2\kappa)x^2}}{\sqrt{2\pi}} \mathrm{d}x \\ &\le 1 + 2\int_{\sqrt{1+z_\star}}^{\infty} \frac{e^{-\frac{1}{2}(1-2\kappa)x^2}}{\sqrt{2\pi}} \mathrm{d}x \\ &\le 1 + \frac{\exp\left(-\frac{z_\star(1-2\kappa)}{2}\right)}{1 - 2\kappa}, \end{aligned}$$

where the last inequality uses some easy algebra and the well known bound on the Gaussian

cdf: $\int_t^\infty \frac{1}{\sqrt{2\pi}} e^{-x^2/2a^2} \mathrm{d}x \le \frac{a^2 e^{-t^2/2a^2}}{t\sqrt{2\pi}}$, valid for all $t > 0$. With $z_\star = c_0 \log p$, We deduce that

$$p \log \mathbb{E}\left[\exp\left(\kappa Z_1^2 \mathbf{1}_{\{Z_1^2 > 1+z_\star\}}\right)\right] \le p \log\left(1 + \frac{p^{-\frac{c_0(1-2\kappa)}{2}}}{(1-2\kappa)}\right) \le \frac{p^{c_0\kappa}}{1-2\kappa}.$$

Hence with $a = \frac{2-2\kappa}{\kappa} \frac{p^{c_0\kappa}}{1-2\kappa} \le 4\kappa^{-1} p^{c_0\kappa}$, (2.18) gives

$$\mathbb{P}\left[\sum_{j=1}^p Z_j^2 \mathbf{1}_{\{Z_j^2 > 1+z_\star\}} > a\right] \le \exp\left(-a\kappa + \frac{p^{c_0\kappa}}{1-2\kappa}\right) \le \exp\left(-p^{c_0\kappa}\right).$$

We conclude that with probability at least $1 - \exp(-p^{c_0\kappa})$, $\sum_{j=1}^p Z_j^2 \mathbf{1}_{\{Z_j^2 > 1+z_\star\}} \le a \le 4\kappa^{-1} p^{c_0\kappa}$, so that

$$\sum_{j \notin \mathsf{I}_{\gamma_\star}} \left(\tilde{\beta}_{n,j} - \beta_{\star,j}\right)^2 \le \frac{4\sigma^2}{n} \kappa^{-1} p^{c_0\kappa}, \tag{2.19}$$

with probability at least $1 - \exp(-p^{c_0\kappa})$. Combining (2.17) and (2.19) it follows that

$$\|\tilde{\beta}_n - \beta_\star\|_2^2 \le \frac{4\sigma^2}{n}\left((1+c_0) s \log p + \frac{p^{c_0\kappa}}{\kappa}\right),$$

with probability at least $1 - \frac{1}{p^{c_0 s/8}} - \exp(-p^{c_0\kappa})$. Finally since $\log s > 1$, we can take $\kappa = \log(s)/(c_0 \log(p)) \in (0, 1/2)$ to achieve $s = p^{c_0\kappa}$. With this choice,

$$\frac{p^{c_0\kappa}}{\kappa} = \frac{s \log(p)}{c_0 \log(s)} \le \frac{s \log(p)}{c_0},$$

and the theorem follows easily. □

2.4.4 Proof of Corollary 1

Proof. Recall that $\mathsf{I} = \mathsf{I}_{\gamma_\star} = \{1 \le j \le p : \beta_{\star,j} \ne 0\}$. We write $u_{\mathsf{I}_{\gamma_\star}} = (u_j,\ j \in \mathsf{I}_{\gamma_\star})$, and $u_{\mathsf{I}^c_{\gamma_\star}} = (u_j,\ j \notin \mathsf{I}_{\gamma_\star})$. It is clear that whenever (2.11) holds and $|\tilde{\beta}_{n,j} - \beta_{\star,j}| \le \sqrt{M\sigma^2 s \log(p)/n}$, we have $\mathsf{sign}(\tilde{\beta}_{n,j}) = \mathsf{sign}(\beta_{\star,j})$. Since $|\tilde{\beta}_{n,j} - \beta_{\star,j}| \le \|\tilde{\beta}_n - \beta_\star\|_2$, we conclude that $\mathsf{sign}(\tilde{\beta}_{n,\mathsf{I}_{\gamma_\star}}) = \mathsf{sign}(\beta_{\star,\mathsf{I}_{\gamma_\star}})$, with probability at least $1 - \frac{1}{p^{(c_0 s)/8}} - \frac{1}{\exp(s)}$.

For the other part, it follows from the definition of $\tilde{\beta}_n$ that for $\beta_{\star,j} = 0$, $\mathsf{sign}(\tilde{\beta}_{n,j}) \ne 0$

implies that $Z_j^2 \geq 1 + z_\star$. Hence

$$\begin{aligned}
\mathbb{P}\Big(\mathsf{sign}(\tilde{\beta}_{n,\mathrm{I}^c_{\gamma_\star}}) \neq \mathsf{sign}(\beta_{\star,\mathrm{I}^c_{\gamma_\star}})\Big) &\leq \sum_{j=1}^{p} \mathbb{P}\Big(Z_j^2 > 1 + z_\star\Big) = \sum_{j=1}^{p} 2\mathbb{P}\Big(Z_j > \sqrt{1+z_\star}\Big) \\
&\leq \sum_{j=1}^{p} e^{-\frac{1}{2}(1+z_\star)} \leq \exp\Big(\log p - \frac{c_0}{2}\log p\Big) \\
&\leq \frac{1}{p^{\frac{c_0}{2}-1}}.
\end{aligned}$$

The results follows. □

2.4.5 Proof of Proposition 3

Proof. We fix an arbitrary $j \in \{1, \ldots, p\}$. We define $x_k = n\gamma_j^{(k)}$, where we omit the dependence on j to keep the notation simple. It follows from (2.13) that

$$x_{k+1} = B\left(\frac{x_k}{\sigma^2 + x_k}\right)^2 + \frac{\sigma^2 x_k}{\sigma^2 + x_k} = \Psi(x_k),$$

where $B = \langle y, X_j\rangle^2/n$, and

$$\Psi(x) = B\left(\frac{x}{\sigma^2 + x}\right)^2 + \frac{\sigma^2 x}{\sigma^2 + x}.$$

Notice that for all $x \geq 0$, $\Psi(x) \in [0, \sigma^2 + B]$. Hence the sequence $\{x_k, k \geq 0\}$ is bounded. The equation $\Psi(x) = x$ is equivalent to $x^2(x + \sigma^2) = x^2 B$. If $B \leq \sigma^2$, $\Psi(x) = x$ has a unique solution $x = 0$. If $B > \sigma^2$, then $\Psi(x) = x$ has two solutions $x = 0$ and $x = B - \sigma^2$. The derivatives of Ψ are given by

$$\Psi'(x) = \frac{\sigma^4}{(\sigma^2 + x)^2} + \frac{2x\sigma^2 B}{(\sigma^2 + x)^3}, \quad \Psi''(x) = \frac{-2\sigma^2 x(\sigma^2 + 2B) + 2\sigma^4(B - \sigma^2)}{(\sigma^2 + x)^4}.$$

We consider two cases

1. Case 1: $B \leq \sigma^2$. Then $\Psi''(x) \leq 0$ for all $x \geq 0$. Hence Ψ is concave. This implies that for all $x \geq 0$,

   $$\Psi(x) \leq \Psi(0) + \Psi'(0)x = x.$$

   This implies that $x_k = \Psi(x_{k-1}) \leq x_{k-1}$. This means that the sequence $\{x_k,\ k \geq 0\}$ is bounded and non-increasing, hence has a limit $x_\star$. By continuity of Ψ, the limit point $x_\star$ satisfies $\Psi(x_\star) = x_\star$. Hence $x_\star = 0$, since we have seen above that 0 is the only fixed-point of Ψ when $B \leq \sigma^2$.

2. Case 2: $B > \sigma^2$: Then $\Psi''(0) > 0$, and by Taylor expansion, in a neighborhood of 0, we have $\Psi(x) \geq \Psi(0) + \Psi'(0)x = x$ for all $x > 0$ small enough. If $x_\star = B_j - \sigma^2$ denotes the unique positive fixed point of Ψ, we can conclude that for all $x \in [0, x_\star]$, $\Psi(x) \geq x$, and for $x > x_\star$, $\Psi(x) < x$. Therefore, if $x_0 \in [0, x_\star]$, then $\{x_k, k \geq 0\}$ is increasing and bounded, hence converges to the unique positive fixed point $x_\star$ (recall that $x_0 > 0$). Whereas, if $x_0 > x_\star$, then $\{x_k, k \geq 0\}$ is decreasing and bounded, hence converges to the unique positive fixed point $x_\star$.

□

CHAPTER 3

Simultaneous Unmixing and Classification of Hyperspectral Images

3.1 Introduction

Hyperspectral imaging in remote sensing increases the spectral resolution of sensors and enables computer object-identification from observed pixels. However, the recovery process remains a challenging task. Indeed, due to the limited spatial resolution of the sensors, each pixel in a hyperspectral image is typically a mixture of few endmember spectral signatures. Hyperspectral unmixing is the task of identifying these mixtures.

The unmixing task can be performed using either a lab-generated library or spectral signatures extracted from the images. Either approach has its own set of assumptions and shortcomings. Lab-generated libraries assumes that the atmospheric distortion does not alter the spectral signature of the endmembers too much while spectral extraction methods assumes that there exists a "pure" pixel in the image. In addition, spectral extraction methods involve solving for the library in addition to the mixture which complicates the task of unmixing further. Some notable spectral extraction methods are N-FINDR [95], VCA [66], and Pixel Purity Index [15]. On the other hand, lab-generated libraries are usually very large in comparison to the spectral resolution of the remote sensor. The high-dimensional library requires the use of regularization methods such as LASSO [87] in order to recover a sparse mixture ([14, 49, 80]).

Mixing models in hyperspectral imaging can be either linear or nonlinear, the linear model being the most commonly used. Nonlinear unmixing appears when there are macroscopic interactions of spectral signatures, which occur typically in multilayered configurations. In this type of scenes, the light scattered by a material is reflected off another material. This type of scenes are commonly seen for example when there is significant tree/plant canopy where light scattered by the canopy is reflected off the ground. Bilinear unmixing models are commonly used for this type of images. There are also other

microscopic interaction of spectral signatures, but in this chapter we focus mainly on the bilinear model. For further details on nonlinear unmixing we point the reader to [34] and the references therein.

Unmixed hyperspectral images in isolation are not easily summarized. In order to increase the interpretability of hyperspectral images, classification methods have been used to classify the pixels. It is standard practice to have pixels belonging to the same class either share the similar mixture or have similar observed spectral signatures. With this in mind, it is common practice to use classification of pixels to introduce spatial dependence between the pixels. This is usually done by introducing Markov random fields. A review of the use of Markov random field in remote sensing can be found in [64]. In another instance of classification, SVM [85] [63] has been applied on observed spectral signatures in order to separate pixels into classes. Besides that, there are Bayesian methods which assumes that the mixture of endmembers in the same group share the same first and second moment [89]. For a review of methods used in hyperspectral image classification, [3, 21] and the references therein are good resources. In addition [51] also has an alternative method for dealing with spatial dependence.

Both unmixing and classification are usually done separately. In this chapter we propose a model that addresses the unmixing and the classification problems jointly, using high-dimensional libraries. Our framework uses a generic library, and can be used with a lab-generated library or a library extracted from the image [69]. In addition, the library can be augmented with nonlinear combinations of the columns in order to cover nonlinear unmixing. The statistical model underpinning our methodology is a regression model driven by a Markov random field, that builds on the work of [89]. However the model developed here is more parsimonious than [89], which makes it possible to apply our method with higher dimensional libraries. Unlike the fully Bayesian approach taken in [89], we adopt the sparse Bayesian learning approach of [88] in order to recover sparse solutions for the mixtures. The method used in this paper requires the use of stochastic EM via MCMC. For further references on stochastic approximation via MCMC, [45] and the references therein are good resources.

3.2 Hyperspectral Unmixing and Classification

3.2.1 The statistical model

We view the hyperspectral data as a three-dimensional data cube with pixels indexed by $p \in \{1, \ldots, P\}$. At a given pixel p, the observed L-spectrum vector $\mathbf{y}_p$ is assumed to contain

a noisy linear mixture of end-members from the library. We make the assumption that each pixel p has a group assignment $z_p \in \{1, \ldots, K\}$, and we make the following distributional assumption:

$$\mathbf{y}_p \sim N(\mathbf{X}\beta_{z_p}, s^2_{z_p}\mathbf{I}_L) \qquad\qquad p = 1, \ldots, P. \tag{3.1}$$

In the above display, $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_R]$ is a known $L \times R$ matrix containing the spectral signatures of the end-members that are present in the library, L is the number of spectral bands, z_p denotes the group label assigned to pixel p, and $\beta_i \in \mathbb{R}^R$, $1 \leq i \leq K$ are parameters. Hence the implicit assumption made in the model is that two pixels with the same group assignment contains the same linear mixture of end-members from the library. However, the observed data at these two pixels might still be different due to the added random noise in (3.1). Also note that the level of noise in each group may be different according to the formulation above. The endmembers $\mathbf{x}_k$ in this case are generic. They may come from lab-generated spectral signatures, spectral signatures extracted from the scene, and/or nonlinear combination of the other endmembers.

Spatial dependence is introduced in the model through the latent group assignments $\{z_p,\ 1 \leq p \leq P\}$ which are assumed spatially dependent. More precisely we assume that the group assignments $\{z_p,\ 1 \leq p \leq P\}$ follows a Potts-Markov random field distribution

$$f(\mathbf{z}|\theta) = \frac{1}{G(\theta)} \exp\left\{ \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta\delta(z_p - z_{p'}) \right\}, \tag{3.2}$$

where $\mathcal{V}(i)$ is the neighborhood (defined here as the four nearest pixel) around pixel i, θ is the granularity parameter that we assume known, $G(\theta)$ is the normalizing constant, $\delta(\cdot)$ is the Kronecker function ($\delta(x) = 1$ if $x = 0$, and 0 otherwise), and $\mathbf{z} = [z_1, \ldots, z_p]$. For further reference on hidden Markov models, [23] is a good reference.

Our model is inspired by, and shares some common features with the model proposed by [89]. However, our formulation is more parsimonious which is key for fast computation. In [89], each pixel has its own coefficient β, and the β's coefficient of the pixels belonging to the same group (same value of z) have the same distribution. This model leads to a very large number of β parameters to estimate, and would not scale well to large images and high-dimensional libraries. We should also mention that the use of the Potts model and other discrete Markov random fields for modeling spatially distributed phenomenon has been explored by other authors (see e.g. [44] and the references therein).

Organizing data for an image with P pixels using standard matrix notation: $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_P]$

and $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_P]$, the likelihood for the data is:

$$f(\mathbf{Y}|z, \boldsymbol{\beta}, s^2) \propto \prod_{p=1}^{P} \left(\frac{1}{s_{z_p}^2} \right)^{L/2} \exp\left\{ -\frac{1}{2s_{z_p}^2} \|\boldsymbol{y}_p - \mathbf{X}\beta_{z_p}\|^2 \right\}.$$

Where $s^2 = (s_1^2, \ldots s_K^2)$ is the different noise levels within the groups. In order to promote sparse solutions for the abundances, we impose on β_k a prior distribution $\pi_{\Sigma_k}(d\beta)$ for some hyper-parameter $\boldsymbol{\Sigma}_k = diag(\sigma_{1,k}, \ldots, \sigma_{R,k})$, where $\sigma_{i,k} \geq 0$. More precisely, we assume the components of β_k are independent, and $\beta_{jk} = 0$ if $\sigma_{jk} = 0$, and $\beta_{jk} \sim N(0, \sigma_{jk}^2)$, if $\sigma_{jk} > 0$. We have identified (and will continue to do so in the sequel) the matrix Σ_k with the vector $(\sigma_{1,k}, \ldots, \sigma_{R,k})$. It should be noted that under this formulation, negative abundances are possible. One possible amendment to the formulation to guarantee nonnegative abundance is to utilize the truncated Gaussian prior instead. Formally, this alternative formulation can be implemented similarly as below, but is computationally more challenging, and would limit the scalability of the proposed approach. See **Section 3.5.1** for further discussion on how we deal with the negative abundances.

With that in mind, we maintained the same form of prior described in **Chapter 2** which yields the joint posterior distribution for both $\boldsymbol{\beta}$ and $\mathbf{z}$

$$\pi(d\boldsymbol{\beta}, \mathbf{z}|\mathbf{Y}, s^2, \boldsymbol{\Sigma}, \theta) \propto f(\mathbf{z}|\theta) f(\mathbf{Y}|z, \boldsymbol{\beta}, s^2) \prod_{k=1}^{K} \pi_{\boldsymbol{\Sigma}_k}(d\beta_k). \tag{3.3}$$

The estimator for the abundances which minimizes risk is the posterior mean of the abundances:

$$\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}(s, \boldsymbol{\Sigma}, \theta) = \iint \boldsymbol{\beta} \pi(d\boldsymbol{\beta}, \mathbf{z}|\mathbf{Y}, s^2, \boldsymbol{\Sigma}, \theta) d\mathbf{z}. \tag{3.4}$$

In (3.4), the integral with respect to $\mathbf{z}$ should be interpreted as a summation. We note that this estimator depends on unknown parameters $\boldsymbol{\Sigma}$, and s^2. We recall that θ is assumed known. Under the full Bayesian treatment, which is the approach taken by [89], further priors are introduced for the hyperparameters. Here we take an empirical Bayes approach, and following the SBL method of [88], we propose to estimate $(\boldsymbol{\Sigma}, s)$ by maximizing the marginal log-likelihood

$$(\hat{\boldsymbol{\Sigma}}, \hat{s^2}) = \arg\min_{\boldsymbol{\Sigma}, s^2} \ell(s^2, \boldsymbol{\Sigma}; \theta, \mathbf{Y}) \tag{3.5}$$

where

$$\ell(s^2, \mathbf{\Sigma}; \theta, \mathbf{Y}) = \log\left[\iint f(\mathbf{z}|\theta) f(\mathbf{Y}|\boldsymbol{z}, \boldsymbol{\beta}, s^2) \prod_{k=1}^{K} \pi_{\mathbf{\Sigma}_k}(d\beta_k) d\boldsymbol{\beta} d\mathbf{z}\right].$$

This problem is a difficult non-concave optimization problem. Building on the EM algorithm of [88, 100], we develop below a Monte Carlo EM algorithm to locate its stationary points. The Monte Carlo EM [93] is a well established method for cases where the computation of the E-step requires Monte Carlo estimates.

Once the hyperparameters $(\hat{\mathbf{\Sigma}}, \hat{s^2})$ are obtained, the estimate for the classification and the mixture is computed by generating MCMC samples from the joint posterior (3.3). The estimate for the classification is set to the posterior mode class for each pixel while the mixture is set to the posterior mean mixture for a given class. For a discussion on the EM algorithm, the seminal paper [33] is a good reference.

3.3 Computation of the hyper-parameters $(\hat{\mathbf{\Sigma}}, \hat{s}^2)$ by Monte Carlo EM

As stated, setting the hyperparameters to the MLE is non-trivial. This necessitates the use of EM algorithm to find an approximate solution. The EM algorithm [33] is an iterative method in finding the maximum likelihood estimate from unwieldy likelihoods. We begin the review of the algorithm by supposing that we observe X with a set of unobserved latent data Z and a set of parameters of interest Θ with complete data likelihood $L(\Theta; X, Z) = f(X, Z|\Theta)$. The maximum likelihood of Θ is obtained by maximizing the data observed likelihood:

$$L(\Theta; X) = \int f(X, z|\Theta) dz$$

In many cases, this likelihood is intractable. The EM algorithm circumvents this problem by defining a surrogate conditional expectation function of the log likelihood in place of the above likelihood:

$$Q(\Theta|\Theta_t) = E_{Z|X,\Theta_t}[\log L(\Theta; X, Z)]$$

This step is called the E-step. After the E-step, we maximize the surrogate function with respect to Θ which is often easier than the original optimization function.

$$\Theta_{t+1} = \arg\max_{\Theta} Q(\Theta|\Theta_t)$$

This is called the M-step. The EM algorithm iterates the E-step and the M-step for a large number of iterations or until a convergence criteria is met. In our case, our implementation begins by treating the marginalized variables $(\boldsymbol{\beta}, \mathbf{z})$ as latent variables. Given $(\theta, \boldsymbol{\Sigma}_t, s_t^2)$, the **complete** data log-likelihood is:

$$\begin{aligned}
\ell_{comp}(\theta, s_t^2, \boldsymbol{\Sigma}_t; \mathbf{Y}, \boldsymbol{\beta}, \mathbf{z}) &= \log\left\{ f(\mathbf{Y}|\boldsymbol{\beta}, s_t^2) f(\boldsymbol{\beta}|\mathbf{z}, \boldsymbol{\Sigma}_t) f(\mathbf{z}|\theta) \right\} \\
&= -\sum_{k=1}^{K} \frac{|\mathcal{I}_k| L}{2} \log s_{k,t}^2 - \sum_{p=1}^{P} \frac{\|\mathbf{y}_p - \mathbf{X}\beta_{z_p}\|^2}{2 s_{z_p,t}^2} \\
&\quad - \sum_{k=1}^{K} \left\{ \frac{1}{2} \log|\boldsymbol{\Sigma}_{k,t}| + \frac{\beta_k^T \boldsymbol{\Sigma}_{k,t}^{-1} \beta_k}{2} \right\} \\
&\quad - \log G(\theta) + \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta \delta(z_p - z_{p'}).
\end{aligned}$$

The surrogate function in the EM algorithm is evaluated by taking expectation of the complete data log-likelihood with respect to the posterior distribution. This gives the E-step of the algorithm:

- E-step:

$$Q_t = \mathbb{E}\left\{ \ell_{comp}(\theta, s_t^2, \boldsymbol{\Sigma}_t; \mathbf{Y}, \boldsymbol{\beta}, \mathbf{z}) | \mathbf{Y} \right\}$$

  Defining the following terms:

$$u_{k,t} = \iint \sum_{p \in \mathcal{I}_k} \|\mathbf{y}_p - \mathbf{X}\beta_k\|^2 \pi(\boldsymbol{\beta}, \mathbf{z}|\mathbf{Y}, s_t^2, \boldsymbol{\Sigma}_t, \theta) d\mathbf{z} d\boldsymbol{\beta} \tag{3.6}$$

$$v_{r,k,t} = \iint \beta_{r,k}^2 \pi(\boldsymbol{\beta}, \mathbf{z}|\mathbf{Y}, s_t^2, \boldsymbol{\Sigma}_t, \theta) d\mathbf{z} d\boldsymbol{\beta} \tag{3.7}$$

$$w = \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \delta(z_p - z_{p'})$$

The surrogate function becomes:

$$Q_t = -\frac{1}{2}\left\{\sum_{k=1}^{K}\left(|\mathcal{I}_k|L\log s_t^2 + \frac{u_{k,t}}{s_{k,t}^2}\right)\right\} - \frac{1}{2}\sum_{k=1}^{K}\sum_{r=1}^{R}\left\{\log\sigma_{r,k,t} + \frac{v_{r,k,t}}{\sigma_{r,k,t}}\right\} - \log G(\theta) + \theta w$$

By maximizing the surrogate function Q_t in (s_t^2, Σ_t), we are solving a simpler problem than the original optimization problem. However, it should be noted that expressions (3.6) and (3.7) which make up the complete surrogate function involves simulating from the joint posterior distribution for the pixel labels and abundances. MCMC is used in order to estimate those three terms. The MCMC used to sample from the distributions are detailed in **Section 3.4.1** . In addition, [27] is a good reference for Gibbs sampling in the context of empirical Bayes. We then use the terms in the M-step of the EM algorithm to update the hyper-paramters.

- M-step:

$$s_{t+1}^2 = \arg\max_{s_t^2} Q_t$$
$$\boldsymbol{\Sigma}_{t+1} = \arg\max_{\boldsymbol{\Sigma}_t} Q_t$$

The solution to the M-steps are:

$$\frac{dQ_t}{ds_{k,t}^2} = -\frac{|\mathcal{I}_k|L}{2s_{k,t}^2} + \frac{u_{k,t}}{2(s_{k,t}^2)^2} \implies s_{k,t+1}^2 = \frac{u_{k,t}}{|\mathcal{I}_k|L} \tag{3.8}$$

$$\frac{dQ_t}{d\sigma_{r,k,t}} = -\frac{1}{2\sigma_{r,k,t}} + \frac{v_{r,k,t}}{2\sigma_{r,k,t}^2} \implies \sigma_{r,k,t+1} = v_{r,k,t} \tag{3.9}$$

The pseudocode for the EM algorithm is outlined below.

repeat

Given s_t^2, $\sigma_{r,k,t}$ and θ

Compute s_{t+1}^2 and $\sigma_{r,k,t+1}$ according to equations (3.8) and (3.9)

u_t and $v_{r,k,t}$ are computed via MCMC simulation.

until convergence

3.4 The MCMC Sampler

3.4.1 Sampling from the posterior $\pi(\beta, z|Y, s^2, \Sigma, \theta)$

Before we go into sampling the hyperparameters, It may be instructive to review Gibb's sampling [28]. Suppose we have a multivariate distribution for random variable $x \in \mathbb{R}^p$ that has the following joint distribution $f(x) = f(x_1, \ldots, x_p)$ and we need to sample n copies from the aforementioned distribution. In addition, we have the full conditional distribution as $f(x_j|x_{-j})$ where $x_{-j} = \{x_1, \ldots, x_{j-1}, x_{j+1}, \ldots x_p\}$. Usually, simulating from the full conditional distribution is much more convenient relative to sampling from the joint distribution. Defining copy i from the $f(x)$ distribution as $x^{(i)} = (x_1^{(i)}, \ldots, x_p^{(i)})$, the Gibb's sampling takes the following form:

1. Initialize $x^{(0)}$.
2. The iterative update successively sample copy $x_j^{(i+1)}$ from the full conditional distribution $f(x_j|x_1^{(i+1)}, \ldots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, \ldots, x_p^{(i)})$. Update $x^{(i+1)}$ is a complete sweep of the marginal variables.
3. Repeat steps 1. and 2. until the required number of samples n.

The sequence $x^{(i)}$ for $i = \{1, \ldots, n\}$ approximates the sample from the joint distribution. The sample from marginal distribution for each x_j is approximated by $x_j^{(i)}$ for $i = \{1, \ldots, n\}$. The mean of the joint distribution can be approximated by averaging all the samples $x^{(i)}$.

As noted earlier, every EM iteration update contains expressions, (3.6) and (3.7), that require MCMC sampling from the posterior distribution $\pi(\boldsymbol{\beta}, \mathbf{z}|\mathbf{Y}, s_t^2, \boldsymbol{\Sigma}_t, \theta)$. Sampling from the non-standard distribution involves Gibbs Sampling which samples from the following full conditional distributions iteratively:

$$\pi(\mathbf{z}|\boldsymbol{\beta}, \mathbf{Y}, s^2, \boldsymbol{\Sigma}, \theta) \propto f(\mathbf{Y}|z, \boldsymbol{\beta}, s^2) f(\mathbf{z}|\theta) \tag{3.10}$$

$$\pi(\beta_k|\mathbf{z}, \boldsymbol{\beta}_{-k}, \mathbf{Y}, s^2, \boldsymbol{\Sigma}, \theta) \sim N(\gamma_k, s_k^2 \Gamma_k) \tag{3.11}$$

Where $\gamma_k = \Gamma_k \mathbf{X}^T \left(\sum_{p \in \mathcal{I}_k} \mathbf{y}_p \right)$ and $\Gamma_k = \left[|\mathcal{I}_k| \mathbf{X}^T \mathbf{X} + s_k^2 \boldsymbol{\Sigma}_k^{-1} \right]^{-1}$. Sampling from the multivariate normal distribution is trivial, but the full conditional for the pixel assignment is much more complex. One approach is to update the each pixel individually, but that method is time consuming and does not scale well for larger images. An alternative/scalable algorithm is outlined in **Section 3.4.2**.

3.4.2 Wolff Clustering Algorithm

The algorithm described in this subsection is commonly applied in the drawing of samples from the Pott's model. In our case, the goal of the algorithm is to draw samples from the following distribution:

$$\pi(\mathbf{z}|\boldsymbol{\beta}, \mathbf{Y}, s^2, \boldsymbol{\Sigma}, \theta) \propto \exp\left\{\sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta\delta(z_p - z_{p'})\right\} \times \prod_{p=1}^{P} \exp\left\{-\frac{\|\mathbf{y}_p - \mathbf{X}\beta_{z_p}\|^2}{2s^2}\right\}$$

Samples from the aforementioned distribution is required in the computation of the E-step of the EM algorithm. We initially applied Gibb's sampler, but found the sampler to be slow for anything larger than 100 pixels. This motivated the use of the Wolff Clustering Algorithm.

The algorithm, first described in [98], involves the use of granularity parameter θ in order to select a cluster with the same class label z. Once the cluster has been determined, a randomly selected proposal label $\tilde{z} \neq z$ for the cluster is compared to the original class label z via the likelihood ratio $\frac{\pi(\tilde{z})}{\pi(z)}$. The probability of the cluster's label being "flipped" to the proposed class label is $\min(1, \frac{\pi_0(\tilde{z})}{\pi_0(z)})$.

This algorithm updates the image classification very efficiently compared to the use of the Gibb's sampler when applied to the drawing of the full posterior distribution for the pixel assignment. The algorithm updates the image assignment by deciding whether to change pixel assignment at the cluster level whereas the Gibb's sampler updates the the pixel assignment at the pixel level. This improves the speed of the EM algorithm and scales better to larger images.

We begin the description of the algorithm by defining the cluster growth probability and the likelihoods used in the comparison of proposed versus original class label. The cluster must be grown systematically using the acceptance probabilities derived from the following:

$$b_{s,l} = \exp\{\theta\}$$

The comparison of the proposed and the original class labels is done using the following

likelihoods:

$$\pi_0(z_1,\ldots,z_{\mathcal{C}_0}) \propto \left[\exp\left\{-\sum_{p\in\mathcal{C}_0}\frac{\|\mathbf{y}_p-\mathbf{X}\beta_{\mathcal{C}_0}\|^2}{2s^2}\right\}\right]$$

$$\frac{\pi_0(\tilde{z})}{\pi_0(z)} = \exp\left\{-\sum_{p\in\mathcal{C}_0}\left(\frac{\|\boldsymbol{y}_p-\boldsymbol{X}\beta_{\tilde{z}}\|}{2s^2_{\tilde{z}}}-\frac{\|\boldsymbol{y}_p-\boldsymbol{X}\beta_{z}\|}{2s^2_{z}}\right)\right\}$$

With the necessary cluster growth probability and likelihoods defined, the algorithm takes the following form:

Given $z=\{z(s), s\in\mathcal{S}\}$, randomly select $s\in\mathcal{C}_0$ and set $\mathcal{C}_0=\{s\}$
Repeat
  For each $l\in\mathcal{C}_0$ and $l'\in\mathcal{D}_l$
    If $z(l)=z(l')$
      Set $\delta_{l,l'}=1$ with probability $1-\frac{1}{b_{s,l}}$
      Set $\delta_{l,l'}=0$ with probability $\frac{1}{b_{s,l}}$
    If $z(l)\neq z(l')$
      Set $\delta_{l,l'}=0$
    Add l' to $\mathcal{C}_0$ if $\delta_{l,l'}=1$
  End for
Until $\mathcal{C}_0$ **can no longer grow**
If $z(s)=i,\ \forall s\in\mathcal{C}_0$
  Randomly select $j\in\{1,\ldots,K\}\backslash\{i\}$ as the proposal $\tilde{z}(s)=j,\ \forall s\in\mathcal{C}_0$
  Accept $\tilde{z}(s)=j,\ \forall s\in\mathcal{C}_0$ with probability $\min(1,\frac{\pi_0(\tilde{z})}{\pi_0(z)})$

3.5 Numerical Experiments

Two simulation studies are performed to evaluate the method's performance. The first involves data sets simulated from a synthetic misspecified model. The second simulation study uses a real world agricultural scene with ground truth classification. For the simulated data experiments, we generate images from the following model.

$$Y_p \sim N\left(X\beta_p, I_{200}\right)$$
$$\beta_p \sim N\left(\mu_{z_p}, \tau^2 I_{500}\right)$$

Where $k=1,\ldots,6$ and $\tau\in\{0.1,0.25,0.5\}$. The mean of the mixtures μ_k are assumed sparse with about 1% nonzero entries coming from a uniform distribution $U(1,2)$. This model is

exactly the same model proposed in [89], and under this model, the β's of pixels sharing the same class share the same first and second moments, but are allowed to be different. This is in contrast to our model where the β's are held constant within a given class. We choose to work with a misspecified model because we would like to examine the performance of the method in the highly likely event that real world conditions present us with diverse intraclass mixtures. The size of the image is 150×150 pixels. We evaluate the performance of the method by computing the misclassification error, and the residual

$$\hat{s}^2 = \frac{\sum_{p=1}^{P} \|\mathbf{y}_p - \mathbf{X}\hat{\beta}_p\|}{L \times P}$$

We performed our estimation using different granularity constants $\theta = \{0.1, 0.3, 0.5\}$. The choice of θ dictates the strength of the spatial correlation when it comes to informing the class of the pixel which influences the mixture of the pixel. Figure 3.1 shows the classification recovery for $\tau = 0.1$. There is negligible difference in the recovery of the classification under different θ conditions. Figure 3.2 shows the classification recovery for $\tau = 0.25$. Once again, there is negligible difference in the recovery of the classification with varying θ. Figure 3.3 shows the classification recovery for $\tau = 0.5$. Under $\tau = 0.5$, the classification recovery rate deteriorates with the increase of θ.

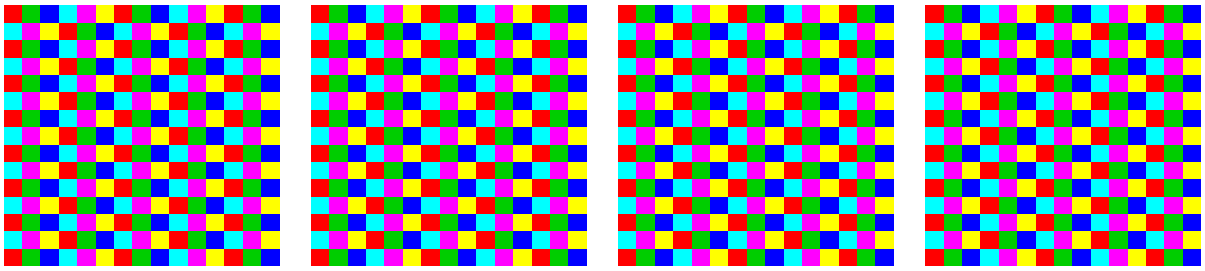


Figure 3.1: The ground truth of the synthetic data is on the top left and the other three plots are the recovered class assignment for different θ with $\tau = 0.1$

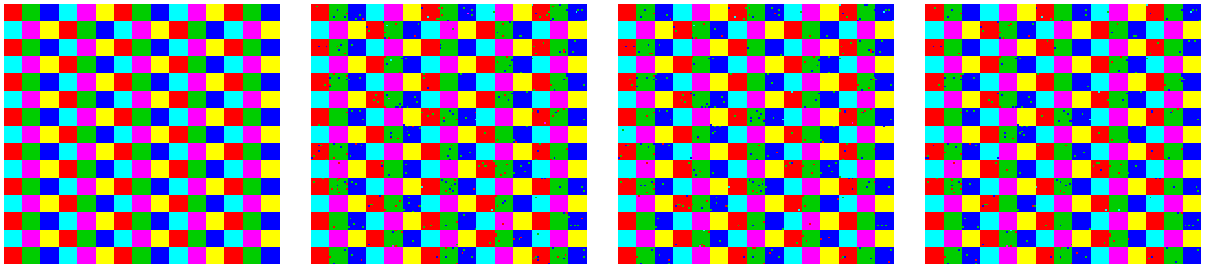


Figure 3.2: The ground truth of the synthetic data is on the top left and the other three plots are the recovered class assignment for different $\theta$with $\tau = 0.25$

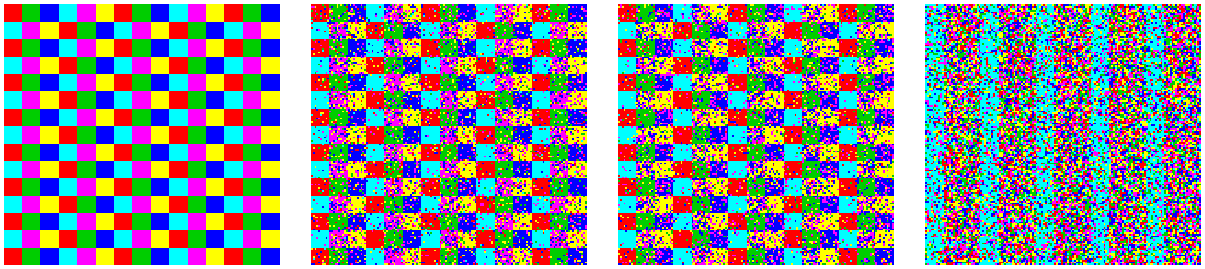


Figure 3.3: The ground truth of the synthetic data is on the top left and the other three plots are the recovered class assignment for different θ with $\tau = 0.5$

The use of different θ levels necessitates the use of residuals of the observed spectral signature in order to determine the optimal θ. In our synthetic simulation, the residuals for low to moderate levels of misspecification, $\tau = \{0.1, 0.25\}$, show very little difference for different levels of spatial correlation θ. However, when the model is highly misspecified, $\tau = 0.5$, there is very little difference in the residuals for $\theta = \{0.1, 0.3\}$ but increased dramatically for $\theta = 0.5$. This implies that a low-to-medium θ is optimal to the recovery of the classification for highly misspecified models. For further reference, Table 3.1 and Table 3.2 are the misclassification rates and residuals for the different scenarios described.

	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$
$\theta = 0.1$	0	0.014	0.226
$\theta = 0.3$	0	0.012	0.296
$\theta = 0.5$	0	0.010	0.758

Table 3.1: Misclassification rate for synthetic data

	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$
$\theta = 0.1$	5.99	32.20	125.66
$\theta = 0.3$	5.99	32.20	125.83
$\theta = 0.5$	5.99	32.20	137.60

Table 3.2: s^2 estimates for synthetic data

Besides selecting θ, we also performed simulation studies on different values of K in our models. In this instance, we performed the study on a synthetic scene with $K = 6$ and about 1% nonzero entries coming from a uniform distribution $U(5, 10)$ with $\tau = 0.1$. We used different values of $K = 3, \ldots, 20$ in our recovery algorithm to see if we could determine the optimal K. The criteria we used to evaluate the suitability of K is BIC:

$$BIC = -2\log f(Y|X,\hat{\beta}) + 2KR \times \log(PL)$$

Figure 3.4 shows a plot of how the BIC changes based on the K used in the recovery algorithm. It is seen from the figure that any increase in the number of groups used in recovery does not decrease the BIC anymore. In fact we observe fluctuations in BIC after the actual $K = 6$ has been reached. This shows that once the model reaches appropriate saturation in terms of number of groups K, any additional parameters will not produce much gain in terms of model fit. Based on this we can conclude $K = 6$ is the optimal number of groups for the recovery algorithm.

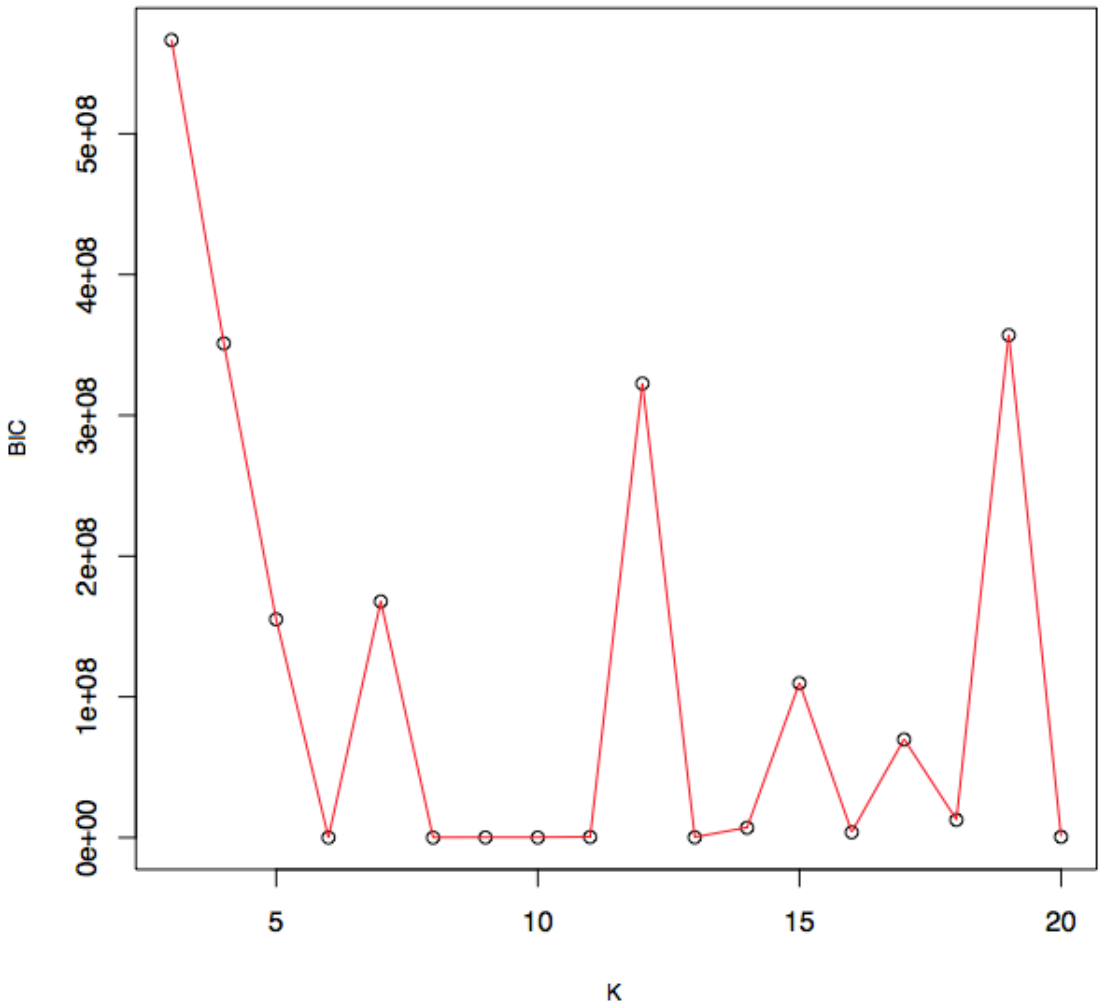


Figure 3.4: Plot of BIC vs the number of classes used in the recovery algorithm.

As a comparison, we also performed the unmixing at the pixel level by solving a constrained optimization problem using alternating direction method of multipliers (ADMM). This method assumes that there is no spatial dependence in the data and fits the data subject to the positivity and additivity constraints. It should be noted that this method results in a solution that has very low residuals, but a high number of parameters (one β for each pixel). The constrained optimization problem takes the form

$$\underset{\beta:\beta_j \geq 0, \sum_j \beta_j = 1}{\operatorname{argmax}} \quad \|y - \mathbf{X}\beta\|$$

Due to the low residuals of the fitted model, we use the penalized residuals of the following form in order to compare this model to the parsimonious model proposed in this chapter. Once again, recall the BIC:

$$BIC = -2\log f(Y|X,\beta) + 2 \times (\text{number of parameters}) \times \log(PL)$$

The results of this comparison is presented in Table 3.3. It is notable that under low and moderate deviations from the proposed model, the model with no spatial dependence performs worse in terms of penalized residuals. However, the non-spatial model is more robust to model misspecification.

	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$
BIC(Spatial)	27,061,609	144,983,014	565,586,619
BIC(Non-Spatial)	345,878,235	348,074,227	355,351,184

Table 3.3: BIC for the spatial and non-spatial model for varying levels of mispecification

3.5.1 Negative entries for β

As noted earlier, under the current formulation, it is possible that the entries of $\boldsymbol{\beta}$ to be negative. This is undesirable because a negative abundance of end-member is not meaningful. In those cases, the researcher has the option of amending the model using the truncated Gaussian instead of the regular Gaussian used in order to ensure non-negativity of the entries. However, this requires significant reworking of the MCMC within the EM iterations which increases the computational complexity. Another approach is to set the negative values to zero. This is the approach that we intend to take in this chapter. From our observations, the method returns negative values that are generally small for scenes with properly matched libraries. Significantly large negative entries suggests the mismatch of the library to the scene as the algorithm reconstructs the observed spectral signature from non-related end-members in the library. As seen in Figure 3.5, even in the case of severe misspecification $\tau = 0.5$, the negative values are still small relative to the signal as long as the library X is appropriate to the scene.

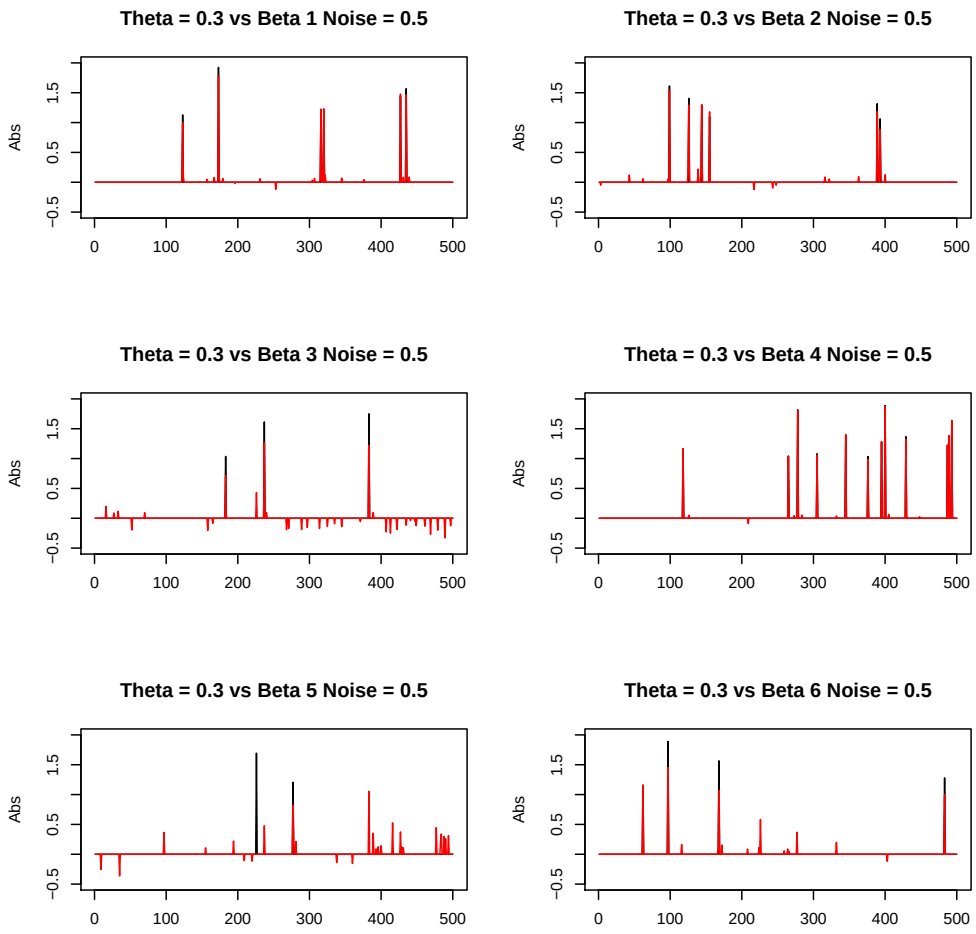


Figure 3.5: Plots of the βs for $\theta = 0.3$ with $\tau = 0.5$. The black lines are the actual signal and the red lines are the actual signal

3.5.2 Indian Pines Scene

In this simulation study, we examine an agricultural scene with ground-truth. The scene comes from Indian Pines which is an agricultural scene with 4 notable zones. The image is a subset of the Indian Pines data set obtained from the website: http://www.ehu.es/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes. We analyze this scene using a nonagricultural United States Geological Survey (USGS) vegetation library taken from the website: http://speclab.cr.usgs.gov/spectral.lib06/). It should be noted that the library used in this case is a mismatched with the scene. It is important to note that crops in different developmental stages exhibit different spectral signatures. For instance, maize when unripened will look green on the visible

spectrum. Ripened maize looks yellow on the visible spectrum. In this example, there is significant difference in terms of the spectral signature on the visible spectrum and non-visible spectrum for the same kind of crop in different stages of development. In addition, remote sensing libraries are created to assist in the survey of areas that are not inhabited and dangerous. An agricultural scene is usually inhabited and there is less motivation to remotely survey agricultural scene due to the relatively inexpensive on-site survey. Even though the library used in this case is mismatched to the scene, we should be able to draw some conclusions regarding the classification even if the abundances recovered in this case is meaningless due to how the model is set up.

The recovery of the classification using lab-generated library is presented in Figure 3.6. Note that the algorithm has collapsed the image into roughly two classes even though we initialized our simulation with 4 classes. In this case, the algorithm is unable to differentiate corn from soybean and considers them as one class. In addition, the algorithm has also collapsed the areas covered by trees, grass and oats into one class.

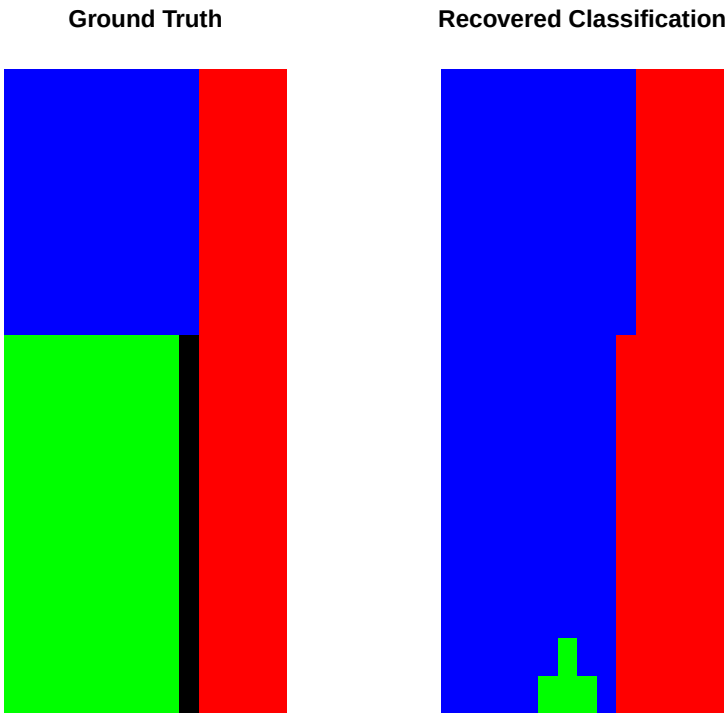


Figure 3.6: Classification plots of the ground truth and the recovered classes. Red is grass and trees, blue is soybean, green is corn, black is oats.

In addition to using the USGS library, we used a library extracted from the Indian Pines image using VCA [66]. VCA is an iterative algorithm that extracts endmembers from observed spectral signature in the image. This method assumes that there is at least one pure pixel in the image. The algorithm takes the following form:

1. Begin by select a random pixel from the image as an endmember in the library. The first endmember can be taken from the pixel with the largest spectral signature.

2. Find the space that is orthogonal to the column space of the library.

3. From the remaining pixels, the pixel with the largest orthogonal projection is added into the library as an endmember.

4. Repeat steps 2 and 3 until an appropriate number of endmembers are included in the library. It is obvious that the maximum number of endmembers cannot exceed the number of pixels in the image.

In this exercise, we extracted 5 endmembers from the image and augmented the 5 endmembers with bilinear combinations resulting in a library containing 20 endmembers. The resulting classification plot is displayed in Figure 3.7. There is very little difference in the classification plots recovered using lab-generated library and VCA recovered nonlinear library. In fact, both libraries are able to differentiate between the grass and trees from the crops fairly well, but are unable to distinguish between different crops. Table 3.4 shows results from the algorithm applied to the lab-generated library and bilinear VCA library for the Indian Pines scene. The results in this table is averaged over twenty repetitions with random initializations for entries of $\mathbf{\Sigma}$ and s^2 drawn from $U(1,5)$

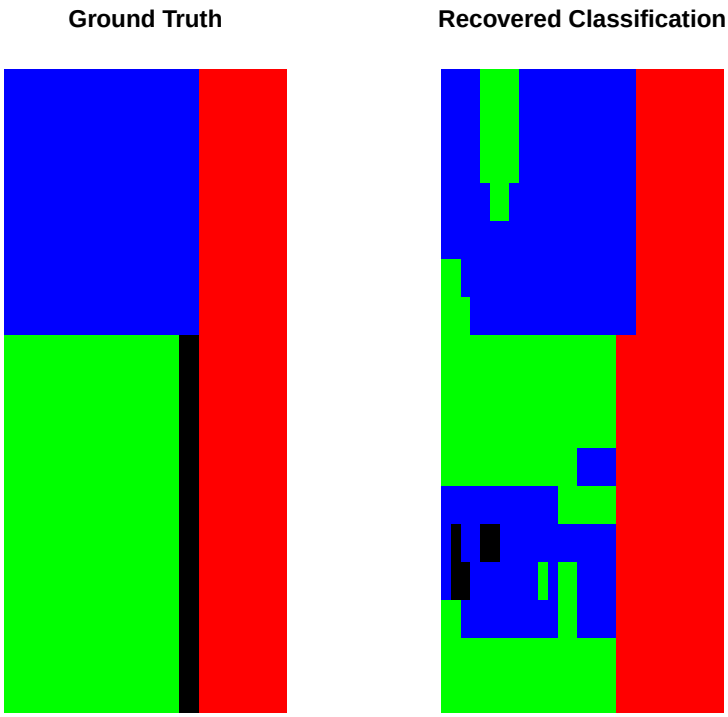


Figure 3.7: Classification plots of the ground truth and the recovered classes. This plot is recovered using the proposed algorithm with VCA recovered endmembers and their bilinear combinations. Red is grass and trees, blue is soybean, green is corn, black is oats.

	Misclassification	$\hat{s}^2$
Lab Library	0.3408	0.0064
VCA Library	0.2404	0.0004
Spectral Clustering	0.2515	N/A

Table 3.4: Misclassification rates and residual $\hat{s}^2$ estimates for classification and unmixing of the Indian Pines scene with lab-generated library and VCA library. The spectral clustering result is recovered using the observed spectral signatures observed from the scene.

As a comparison, we conducted spectral clustering in order to provide a data driven approach to classify the pixel based on the l_2 distance between the spectral signature. The

spectral clustering method builds an adjacency matrix W using the following kernel:

$$W_{ij} = \exp\left\{-\frac{||y_i - y_j||_2^2}{L}\right\}$$

From the adjacency matrix W we construct the Laplacian matrix U:

$$D = diag\left(\sum_j W_{ij}\right)$$
$$U = D - W$$
$$L_n = UD^{-1}$$

where L_n is the normalized Laplacian. Once this has been constructed, we apply the eigen-decomposition to L_n.

The number of groups K is determined by examining the smallest eigenvalues. The eigenvalues of L_n are mostly close to 1. Suppose the eigenvalues of L_n are arranged in ascending order $\{e_{(1)}, \dots e_{(P)}\}$

$$K = min\{i : (1 - e_{(i)}) \geq \tau, i = 1, \dots P\}$$

Where τ is the threshold for setting the number of groups. This threshold is dependent on the observed data. Setting this requires a certain amount of judgement regarding the number of groups compared to the number of pixels in an image.

Once K has been established, we collect K eigenvectors associated with the K smallest eigenvalues into the matrix $V \in \mathbb{R}^{P \times K}$. We then apply K-means clustering on the P K-dimensional row-vectors.

Figure 3.8 shows the classification results from spectral clustering. This is recovered using just the distance between observed spectral signatures. From the plots, we can see that the resulting classification is consistent with the results using the proposed method. The difference in spectral signature between the crops are too small to differentiate between them. This classification method is also able to differentiate between the non-crop pixels and the crop pixels reasonably well. This validates the results recovered using the proposed method.

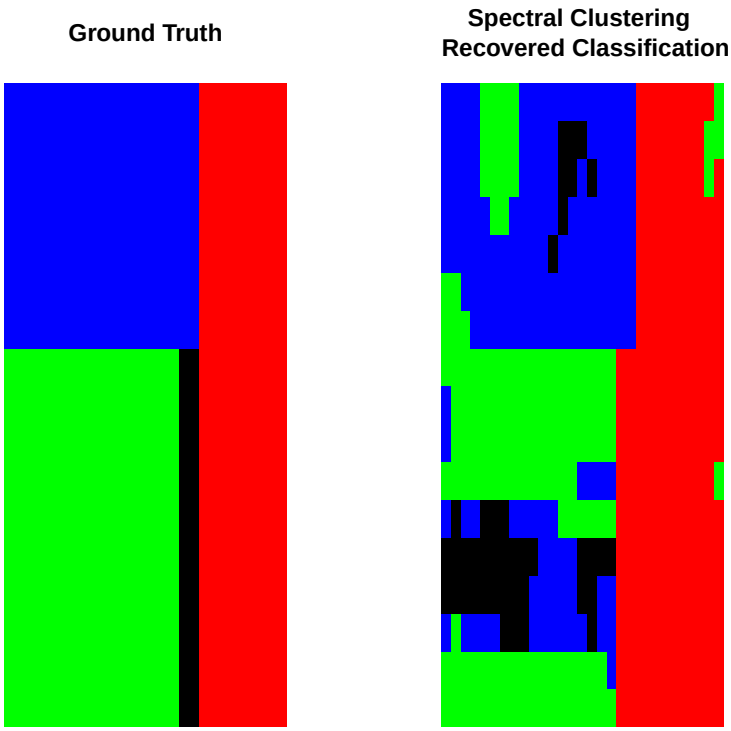


Figure 3.8: Classification plots of the ground truth and the recovered classes. This plot is recovered using spectral clustering. Red is grass and trees, blue is soybean, green is corn, black is oats.

3.6 Conclusion

The main contribution of this chapter is the application of the unmixing and classification of hyperspectral images simultaneously. As mentioned earlier, most preexisting literature approach to this as a two-stage problem. The classification of the images also introduces spatial dependence through the Potts-model parameter θ. In our synthetic data simulations, the greater the spatial dependence, the less robust the method is to model misclassification. However, in real data simulations, the results of the classification is encouraging. The results of the unmixing is only relevant if the library is well matched with the scene. As seen in [100], the unmixing utilizing SBL is dependent on the strength of the signal relative to the noise and the maximal correlation between the columns of the library. If the lab-generated library is not available, the library of spectral signatures can be extracted from

the scene and augmented with nonlinear combinations to account for nonlinear mixtures. This shows the flexibility of the model in dealing with nonlinear unmixing in a scene.

CHAPTER 4

Hyperspectral Unmixing with Wavelength Dependence

4.1 Introduction

Hyperspectral images have become ubiquitous due to recent technological advancement in imaging technology. These images typically contain thousands of pixels with each pixel containing a vector of observed spectral signature. The length of these vectors depends on the spectral resolution of the sensor. The area represented by each pixel depends on the spatial resolution of the sensor. The task of unmixing is incredibly complex because the low spatial resolution means each pixel contains a mixture of endmembers in addition to attenuation of the signal reaching the sensor.

These challenges spawned extensive literature on the subject. Some literature [63, 64, 85, 89] incorporates spatial dependence in order to classify and/or unmix the data. Spatial correlation improves the unmixing of the data due to pixels being grouped together contain similar mixing properties. Most of the literature makes use of Markov random fields in order to classify the pixels. Pixels belonging to the same class usually share the same first and/or second moment properties [89].

Some methods make use of lab-generated libraries [50, 101] instead of pixel derived libraries [66,89,95] in order to unmix the data. Due to the size of lab-generated libraries, the problem becomes a high-dimensional (number of parameters $\geq$ number of samples) one. This necessitates that the method recover a sparse solution to the mixture problem [50]. In some cases, the recovery of the library from the image is as important as recovery of the mixture [77, 92].

Wavelength dependence has not been as well studied as spatial dependence in the context of hyperspectral unmixing. To the best of our knowledge, there is no existing literature that incorporates wavelength dependence in hyperpectral modeling. In this chapter we try

to address both spatial and spectral wavelength dependence in the unmixing of hyperspectral data. The endmember library used in this chapter is generic: the library can be linear or nonlinear. In the real data simulation, the VCA-recovered [66] library can be augmented to accommodate nonlinear (bilinear) [34, 67] endmember mixtures. We introduce spatial dependence by classifying the pixels using spectral clustering. Within each class of pixels, we proceed to unmix the pixels while estimating spectral wavelength dependence. To motivate the inclusion of wavelength dependence, we included **Figure 4.1** of three randomly selected pixels from a scene (Reno) used in this chapter. From the plots, it is obvious that the spectral signatures display wavelength dependence due to the smooth portions of the curves. Therefore, assuming that there is minimal absorbtion in contiguous wavelength bands, the spectral signature should be smooth. We used the concentration matrix in the multivariate gaussian to model the dependence among the wavelengths. The concentration of the multivariate gaussian is appealing because a zero entry in the off-diagonal of the concentration matrix implies conditional independence as shown in **Section B.1**. We intend to recover somewhat sparse concentration matrix which will contain non-zero entries only for the most correlated wavelength bands.

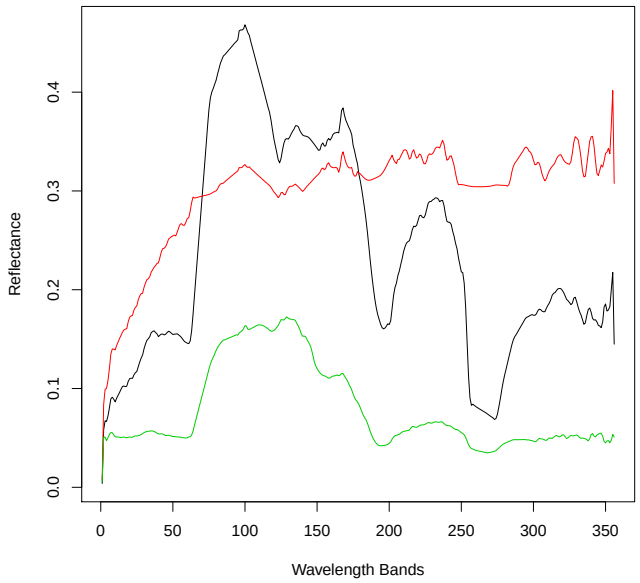


Figure 4.1: Plots of spectral signature for 3 randomly selected pixels from Reno

4.2 The Unmixing Model

The unmixing model as described in this section makes use of a generic library X which has columns denoting the actual spectral signature of endmembers and/or the spectral signature of nonlinear combinations of the original endmember spectral signatures. We assume that each group of pixels have the same linear mixture of endmembers. Specifically, a pixel in a given group has the following linear form:

$$Y_i = X\beta + \epsilon$$

where $\epsilon \sim N(0, \Sigma)$, $Y_i \in \mathbb{R}^L$, $X \in \mathbb{R}^{L \times R}$, and $\beta \in \mathbb{R}^R_+$. The library X is recovered within each group of pixels using VCA from [66]. β dictates the mixture of end-members for the group. In most applications, the mixture is determined at the pixel level. However, our approach assumes the pixels has been grouped in the manner described in **Section 4.4** which has put pixels with similar observed spectral signatures into groups. This is done in order to simplify the formulation by assuming that pixels belonging to the same group have the same endmember mixture β. In addition, the mixture is governed by the following constraints:

$$\sum_{j=1}^{R} \beta_j = 1, \qquad\qquad \beta_j \geq 0$$

The two constraints are known as the additivity and positivity constrain respectively which is standard in hyperspectral unmixing literature [14, 49, 50, 92]. With this formulation, the likelihood of the model takes the form of a multivariate gaussian with a non-diagonal covariance matrix:

$$f_{\beta,\Sigma}(Y) = \prod_{i=1}^{N} (2\pi)^{-\frac{L}{2}} |\Sigma^{-1}|^{\frac{1}{2}} \exp\left\{ -\frac{1}{2} (Y_i - X\beta)^T \Sigma^{-1} (Y_i - X\beta) \right\}$$

Typically, the covariance matrix in this model is diagonal, implying that there is no wavelength correlation. In practice, we frequently observe spectral signatures that are smooth as seen in **Figure 4.1**. This means the adjacent wavelength bands are correlated implying some of the off-diagonal values of the concentration matrix are non-zero. Suppose we define the concentration matrix $\Theta = \Sigma^{-1}$ and the empirical covariance matrix

$S(\beta) = \frac{1}{N}\sum_{i=1}^{N}(Y_i - X\beta)(Y_i - X\beta)^T$, the negative log-likelihood takes the form:

$$\ell(\beta, \Theta) = -\frac{1}{2}\log|\Theta| + \frac{1}{2}tr\{\Theta S(\beta)\}$$

We perform graphical LASSO and hyperspectral unmixing by minimizing the function above subject to some constraint/regularization on the variables Θ and β. With the addition of the constraints/regularization, the constrained optimization problem becomes:

$$(\hat{\beta}, \hat{\Theta}) = \arg\min_{\Theta,\beta} -\frac{1}{2}\log|\Theta| + \frac{1}{2}tr\{\Theta S(\beta)\} + \sum_{i,j}\alpha\lambda|\theta_{ij}| + \frac{1-\alpha}{2}\lambda\theta_{ij}^2$$

$$\text{s.t. } \beta_j \geq 0,\ \sum_{j=1}^{R}\beta_j = 1$$

Where $Reg_\lambda(\Theta) = \sum_{i,j}\alpha\lambda|\theta_{ij}| + \frac{1-\alpha}{2}\lambda\theta_{ij}^2$ with $\alpha \in [0,1]$ is the constraints/regularization imposed on Θ to to promote sparsity in the concentration matrix Θ which is desirable due to the conditional independence of the wavelengths implied by the zero entries. In addition, the constraints of β is imposed to ensure the positivity of the entries and to ensure the entries sum up to 1. This optimization problem is bi-convex, but solving for both Θ and β parameters simultaneously is challenging due to the constraints imposed on both parameters.

4.3 Computation

Solving the optimization as defined in the previous section is difficult. However, we solve this by breaking the complex constrained optimization problem into blocks of simpler constrained optimization problems [99]. The blocks in question are the β and Θ blocks. Suppose we define the unconstrained optimization function as

$$L(\Theta,\beta) = -\frac{1}{2}\log|\Theta| + \frac{1}{2}tr\{\Theta S(\beta)\} + \sum_{i,j}\alpha\lambda|\theta_{ij}| + \frac{1-\alpha}{2}\lambda\theta_{ij}^2$$
$$+ m(\beta^T 1_R - 1) + \frac{\lambda_m}{2}(\beta^T 1_R - 1)^2 + I_+(\beta)$$

Where $Reg_{\lambda_\Theta}(\Theta) = \sum_{i,j}\alpha\lambda|\theta_{ij}| + \frac{1-\alpha}{2}\lambda\theta_{ij}^2$ and $Reg_{\lambda_\beta}(\beta) = m(\beta^T 1_R - 1) + \frac{\lambda_m}{2}(\beta^T 1_R - 1)^2 + I_+(\beta)$ are appropriate regularization/Lagrangian which incorporate the constraints for Θ and β respectively. The block coordinate descent algorithm alternately optimizes the objective

function:

$$\hat{\beta}^{(t+1)} = \arg\min_{\beta} L(\Theta^{(t)}, \beta) \tag{4.1}$$

$$\hat{\Theta}^{(t+1)} = \arg\min_{\Theta} L(\Theta, \beta^{(t+1)}) \tag{4.2}$$

If the objective function $L(\Theta,\beta)$ is convex, the block coordinate updates will converge to the optimal solution [99]. The unconstrained optimization problem (augmented Lagrangian) for (4.1) takes the form:

$$\arg\min_{\beta} \frac{1}{2}\sum_{i=1}^{N}(Y_i - X\beta)^T\Theta(Y_i - X\beta) + m(\beta^T 1_R - 1) + \frac{\lambda_m}{2}(\beta^T 1_R - 1)^2 + I_+(\beta)$$

Meanwhile, the unconstrained optimization problem for (4.2) takes the form

$$\arg\min_{\Theta} -\log\det\Theta + tr(\Theta S) + \sum_{i,j}\alpha\lambda|\theta_{ij}| + \frac{1-\alpha}{2}\lambda\theta_{ij}^2$$

With $\alpha \in [0,1]$. Under the special case where $\alpha = 1$, the problem is analogous to the LASSO [87] applied to the estimation of the concentration matrix. The addition of the second term corresponding to $\alpha \neq 1$ is comparable to the elastic net [104] to aid in terms of variable selection. The regularization parameter λ in conjunction with α regulates the level of sparsity of the concentration matrix.

However, due to the complexity of (4.1) and (4.2) blocks, we adopt an modified version of the block coordinate descent. Under normal block coordinate descent, the optimization blocks (4.1) and (4.2) have to be solved directly in each iteration. Our modification will only solve the two blocks approximately by iteratively applying one cycle of the updates in **Section 4.3.1** and **Section 4.3.2** at each step of the algorithm. The algorithm outlined as pseudocode is in **Section 4.3.3**

4.3.1 The β Updates

The β optimization block can be solved using ADMM [5, 18] . In order to provide context on how ADMM works, it is useful to go over the building blocks of the method. Suppose we have the following equality-constrained optimization problem:

$$\text{minimize } f(x) \text{ subject to } Ax = b$$

Where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $f(x) \in \mathbb{R}$ is convex. The Lagrangian of the problem takes the form:

$$L(x, y) = f(x) + y^T(Ax - b)$$

The dual ascent method consists of iterating the following updates:

$$x^{(s+1)} = \arg\min L(x, y^{(s)})$$
$$y^{(s+1)} = y^{(s)} + \alpha^{(s)}(Ax^{(s+1)} - b)$$

Intuitively the dual ascent method minimizes the function with the first step while it updates the "price" for violating the constraints $Ax = b$ in the second step. With an appropriate choice of $\alpha^{(s)}$ and some regularity conditions/assumptions hold, the dual ascent method minimizes the function f while increasing the "price" for violating the constraints thus ensuring the the solution solves the constrained optimization function. [31, 32] are good references on the dual ascent method.

However, the assumptions necessary for the convergence of the dual ascent method do not hold in many applications (f is not strictly convex or f is not bounded above). Therefore, augmented Lagrangian methods were developed in order to address this issue and make the dual ascent method more robust. The augmented Lagrangian of the aforementioned constrained optimization is:

$$L_\rho(x, y) = f(x) + y^T(Ax - b) + \frac{\rho}{2}||Ax - b||^2$$

The iterative method for solving this is called the Method of Multipliers that takes the following form:

$$x^{(s+1)} = \arg\min L(x, y^{(s)})$$
$$y^{(s+1)} = y^{(s)} + \rho(Ax^{(s+1)} - b)$$

Where ρ is used as step-size updates in place of $\alpha^{(s)}$. The method of multipliers converges under far more general conditions including cases when f is not convex and unbounded from above. In order to motivate the choice of ρ, let's suppose f is differentiable. The optimality conditions for the constrained optimization is primal and dual feasibility:

$$Ax^\star - b = 0$$
$$\nabla f(x^\star) + A^T y^\star = 0$$

By definition, if $x^{(s+1)}$ minimizes $L_\rho(x, y^{(s)})$,

$$\begin{aligned} 0 &= \nabla_x L_\rho(x^{(s+1)}, y^{(s)}) \\ &= \nabla_x f(x^{(s+1)}) + A^T(y^{(s)} + \rho(Ax^{(s+1)} - b)) \\ &= \nabla_x f(x^{(s+1)}) + A^T y^{(s+1)} \end{aligned}$$

Which means the pair $(x^{(s+1)}, y^{(s+1)})$ is dual feasible. The further along the algorithm, the smaller the primal residual $Ax^{(s+1)} - b$ becomes, thus ensuring primal and dual feasibility which results in an optimal solution. [12, 48] are good references on the method of multipliers.

ADMM builds on the decomposability of dual ascent and convergence properties of method of multipliers. In order to see that, suppose we have the following optimization problem

$$\text{minimize } f(x) + g(z) \text{ subject to } Ax + Bz = b$$

Where $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$, $A \in \mathbb{R}^{p \times n}$, and $B \in \mathbb{R}^{p \times m}$, and $c \in \mathbb{R}^p$. Assuming that f and g are convex, the resultant augmented Lagrangian of the optimization problem is:

$$L_\rho(x, y.z) = f(x) + g(z) + y^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|^2$$

The crucial difference between this formulation compared to the earlier formulation is the splitting of variable x from earlier into two parts x and z. Solving the Lagrangian using ADMM involves the following iterations:

$$\begin{aligned} x^{(s+1)} &= \arg\min_x L_\rho(x, z^{(s)}, y^{(s)}) \\ z^{(s+1)} &= \arg\min_x L_\rho(x^{(s+1)}, z, y^{(s)}) \\ y^{(s+1)} &= y^{(s)} + \rho(Ax^{(s+1)} + Bz^{(s+1)} - c) \end{aligned}$$

ADMM as outlined above is very similar to the dual ascent and method of multipliers updates outlined earlier, but with a crucial difference. Under the method of multipliers the update $(x^{(s+1)}, z^{(s+1)})$ are jointly found while ADMM "alternates" between the two variables in its updates.

The constraints on β lead naturally to the formulation of the augmented Lagrangian [92]. In this section, we estimate β via one iteration of ADMM at each cycle of the main algorithm in **Section 4.3.3**. Building on the formulation outlined, we introduce the aug-

mented Lagrangian in order to convert the constrained optimization problem to an unconstrained one:

$$L(\beta) = \frac{1}{2}\sum_{i=1}^{N}(Y_i - X\beta)^T\Theta(Y_i - X\beta) + m(\beta^T 1_R - 1) + \frac{\lambda_m}{2}(\beta^T 1_R - 1)^2 + I_+(\beta)$$

The first term governs the fit of the model while the second and third term represents the additivity constraint and its augmented Lagrangian. The final term is the positivity constraint where:

$$I_+(\beta)_j = \begin{cases} 0 & \text{if } \beta_j \geq 0 \\ \infty & \text{if } \beta_j < 0 \end{cases}$$

This constraint limits the solution space of the β to the positive quadrant of the $\mathbb{R}^R$ space. In order to implement ADMM, we implement an auxiliary variable u in order to split the optimization problem. With the introduction of the auxiliary variable, the optimization problem becomes:

$$\begin{aligned} L(\beta) = &\frac{1}{2}\sum_{i=1}^{N}(Y_i - X\beta)^T\Theta(Y_i - X\beta) + m(\beta^T 1_R - 1) + \frac{\lambda_m}{2}(\beta^T 1_R - 1)^2 \\ &+ q^T(\beta - u) + \frac{\lambda_q}{2}\|\beta - u\|^2 + I_+(u) \end{aligned}$$

Assuming Θ is known, solving this optimization problem results in the following iterative updates:

$$\begin{aligned} \beta^{k+1} &= \left(NX^T\Theta X + \lambda_m^k 1_R 1_R^T + \lambda_q I_R\right)^{-1}\left(X^T\Theta(N\bar{Y}) - m^k 1_R + \lambda_m^k 1_R - q^k + \lambda_q u^k\right) \\ u^{k+1} &= \left[\frac{q^k 1_R + \lambda_q \beta^{k+1}}{\lambda_q}\right]^+ \\ m^{k+1} &= m^k + \lambda_m(1_R^T\beta^{k+1} - 1) \\ q^{k+1} &= q^k + \lambda_q(\beta^{k+1} - u^{k+1}) \\ \lambda_m^{k+1} &= \rho_m\lambda_m^k \\ \lambda_q^{k+1} &= \rho_q\lambda_q^k \end{aligned} \tag{4.3}$$

These updates are repeated once in each iteration in the main algorithm in **Section 4.3.3**. These updates are the β block update in (4.1). This represents part of one iteration in **Section 4.3.3**. This update in conjunction with (4.6) form one complete iteration in **Section**

4.3.3.

4.3.2 The Θ Updates

In this section, we address the presence of wavelength dependence in our model. Assuming β is known in this case, solving for the concentration matrix $\Theta = \Sigma^{-1}$ is equivalent to solving the following optimization problem:

$$\hat{\Theta} = \arg\min_{\Theta} -\log\det\Theta + tr(\Theta S) + Reg_{\lambda}(\Theta)$$

Recall $S = \frac{1}{N}\sum_{i=1}^{N}(Y_i - X\beta)(Y_i - X\beta)^T$ is the empirical covariance matrix and $Reg_{\lambda}(\Theta) = \sum_{i,j}\alpha\lambda|\theta_{ij}| + \frac{1-\alpha}{2}\lambda\theta_{ij}^2$. The regularization term in the optimization problem is imposed in order to induce a sparse concentration matrix and to ensure the recovered concentration has a large condition number. This is desirable because zero entries in the concentration matrix implies conditional independence of the respective wavelength bands as a result of the multivariate normal distribution. **Section B.1** contains a simple proof of this fact. Suppose we define the optimization such that:

$$\hat{\Theta} = \arg\min_{\Theta} f(\Theta) + g_{\lambda}(\Theta) \tag{4.4}$$

where $f(\Theta) = -\log\det\Theta + tr(\Theta S)$ and $g_{\lambda}(\Theta) = Reg_{\lambda}(\Theta)$. In addition, we define the proximal operator [35, 72] on the function h as

$$prox_{\psi h}(v) = \arg\min_{x}\left\{h(x) + \frac{1}{2\psi}\|x - v\|^2\right\}$$

The intuition behind the proximal operator $prox_{\psi h}(v)$ is to minimize the function h while being in the proximity of v while the scaling parmeter ψ is the trade-off between minimizing the function h and being close to v. An alternate interpretation of the the proximal operator, under certain assumptions [72], takes the form of a gradient update $prox_{\psi h}(v) \approx v - \nabla h(v)$.

Using the proximal operator, the solution to optimization problem (4.4) can be solved by recursively applying the following update:

$$\Theta_{k+1} = prox_{\delta_k g}\{\Theta_k - \delta_k \nabla f(\Theta_k)\} \tag{4.5}$$

The update is repeated until convergence. If $g(\Theta) = Reg_{\lambda}(\Theta) = \sum_{i,j}\alpha\lambda|\theta_{ij}| + \frac{1-\alpha}{2}\lambda\theta_{ij}^2$, the

proximal operator has this simple form [9]:

$$prox_{\psi g}\{\Theta\} = \arg\min_t \left\{ g(x) + \frac{1}{2\psi}||t-\Theta||^2 \right\}$$

$$\left(prox_{\psi g}\{\Theta\}\right)_{ij} = \begin{cases} 0 & \text{if } |\theta_{ij}| < \alpha\lambda\psi \\ \frac{\theta_{ij}-\alpha\lambda\psi}{1+(1-\alpha)\lambda\psi} & \text{if } \theta_{ij} \geq \alpha\lambda\psi \\ \frac{\theta_{ij}+\alpha\lambda\psi}{1+(1-\alpha)\lambda\psi} & \text{if } \theta_{ij} \leq -\alpha\lambda\psi \end{cases}$$

Note that $\nabla f(\Theta) = S - \Theta^{-1}$. Therefore (4.5) takes the form:

$$\begin{aligned} (\Theta_{k+1})_{ij} &= \left(prox_{\delta_k g}\{\Theta_k - \delta_k \nabla f(\Theta_k)\}\right)_{ij} \\ &= \left(prox_{\delta_k g}\{\Theta_k - \delta_k (S - N\Theta_k^{-1})\}\right)_{ij} \\ &= \begin{cases} 0 & \text{if } |[\Theta_k - \delta_k(S-\Theta_k^{-1})]_{ij}| < \alpha\lambda\delta_k \\ \frac{[\Theta_k - \delta_k(S-\Theta_k^{-1})]_{ij} - \alpha\lambda\delta_k}{1+(1-\alpha)\lambda\delta_k} & \text{if } [\Theta_k - \delta_k(S-\Theta_k^{-1})]_{ij} \geq \alpha\lambda\delta_k \\ \frac{[\Theta_k - \delta_k(S-\Theta_k^{-1})]_{ij} + \alpha\lambda\delta_k}{1+(1-\alpha)\lambda\delta_k} & \text{if } [\Theta_k - \delta_k(S-\Theta_k^{-1})]_{ij} \leq -\alpha\lambda\delta_k \end{cases} \end{aligned} \tag{4.6}$$

These updates are repeated once in each iteration in the main algorithm in **Section 4.3.3**. This represents the update for the Θ block update in (4.2). This represents half of one iteration of the pseudocode in **Section 4.3.3**. This update in conjunction with (4.3) form one complete iteration of the pseudocode in **Section 4.3.3**.

4.3.3 Pseudocode

The algorithm combines updates (4.3) and (4.6) at each iteration to form one complete iteration of the updates to the parameters of interest (β, Θ). Also note that this algorithm combines one update step from ADMM and proximal gradient algorithm which updates the parameters in tandem. The pseudocode is as follows:

repeat

Given $(\beta^k, u^k, m^k, q^k, \lambda_m^k, \lambda_q^k)$ and Θ^k

Update $(\beta^{k+1}, u^{k+1}, m^{k+1}, q^{k+1}, \lambda_m^{k+1}, \lambda_q^{k+1})$ according to updates in (4.3)

while holding Θ^k constant.

Update Θ^{k+1} according to updates derived in (4.6)

while holding β^{k+1} constant.

until convergence

4.4 Spectral Clustering

While the wavelength dependence/correlation is parameterized explicitly in the unmixing model, the spatial dependence is a bit more subtle. Assuming that the hyperspectral image can be partitioned into G number of groups in which pixels within the group share a library and each pixel is a linear combination of the endmembers in the library as defined in **Section 4.2**. We model the spatial dependence by grouping "similar" pixels into groups. We assume the pixels belonging to the same group contain the same mixture parameter β and wavelength dependence parameter Θ. We begin by classifying the hyperspectral image into separate groups using spectral clustering [58]. Spectral clustering is used to partition the pixels into groups. Suppose there are P pixels in the image, the method begins by building an adjacency matrix $W \in \mathbb{R}^{P\times P}$ using the following kernel:

$$W_{ij} = \exp\left\{-\frac{\|y_i - y_j\|_2^2}{L}\right\}$$

where y_i is the spectral signature observed at pixel i. The kernel used here only depends on the distance between the spectral signatures. The kernel may be altered to incorporate various measures which can be used to quantify how "similar" the pixels are hence W is often referred to as the similarity or adjacency matrix, but for the purpose of this paper we will concentrate on the distance between the spectral signatures. Once the adjacency matrix has been constructed, we proceed to build the Laplacian matrix U:

$$D = diag\left(\sum_j W_{ij}\right)$$
$$L_n = D^{-1/2} W D^{-1/2}$$

where L_n is the normalized Laplacian. Once this has been constructed, we apply the eigen-decomposition to L_n. It is important to note that there are alternate ways to define the Laplacian. In this formulation, the leading eigenvalues are most pertinent, while the formulation as in [58] will result in the case where the smallest eigenvalues are most meaningful to spectral clustering. The number of groups G is determined by examining the largest eigenvalues. Suppose the eigenvalues of L_n are arranged in descending order $\{e_{(P)}, \dots e_{(1)}\}$, the number of groups is:

$$G = \max\{P - i : e_{(i)} \geq \tau, i = 1, \dots P\}$$

Where τ is the threshold for setting the number of groups. At this point, it is important to note that spectral clustering is not an unsupervised approach to clustering. The threshold τ is dependent on the observed data and setting this requires a certain amount of judgement regarding the number of groups relative to the number of pixels in the image.

Once G has been established, we collect G eigenvectors associated with the G smallest eigenvalues into the matrix $V \in \mathbb{R}^{P \times G}$. We then apply K-means clustering on the P G-dimensional row-vectors. Once the classes of the pixels have been established, we then perform hyperspectral unmixing with graphical LASSO to the data within each group of pixels. **Figure 4.2** is an example of a classification plot recovered from a synthetic $G = 4$ problem. For intuition on spectral clustering, please refer to **Section B.2**.

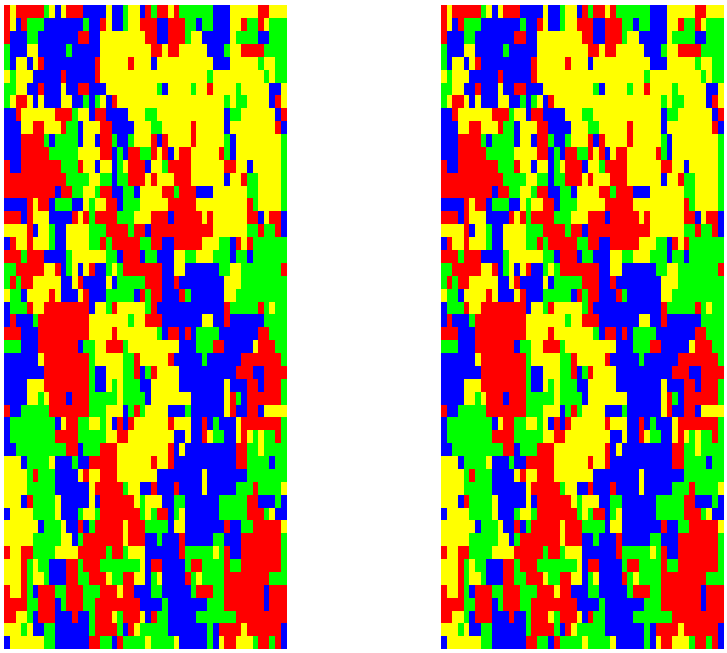


Figure 4.2: Classification plots of the ground truth (left) and the recovered (right) classes for synthetic data using exact spectral clustering. Although this method is extremely accurate in terms of classification, it does not scale for larger images due to memory and computation constraints placed in storing large exact adjacency matrix and diagonalizing it.

4.4.1 Introduction to the Nystrom Method

Before presenting the Nystrom method as applied in spectral clustering, we review the Nystrom method as a general method designed to approximate solutions to integral equations. Specifically, we are interested in integral equations of the form:

$$\int_0^1 W(x,y)\phi(y)dy = \lambda\phi(y)$$

Where $x, y \in \mathbb{R}$, $\phi(x)$ represents the eigenfunction, and $W(x,y)$ is the similarity between x and y. The goal in this case is to solve for the eigenvalues λ and eigenvectors $\phi(\cdot)$ that satisfy the integral equation. The Nystrom method approximates the integral equation as:

$$\frac{1}{m}\sum_{i=1}^{m} W(x,y_i)\hat{\phi}(y_i) = \hat{\lambda}\hat{\phi}(x)$$

Where $y_1, \ldots, y_m$ are the $m \ll n$ sampled points from the data, $\hat{\lambda}$ and $\hat{\phi}(x)$ are the the approximate eigenvalue and approximate eigenfunction respectively. Suppose we define the $\tilde{W}_{ij} = W(y_i, y_j)$ as the similarity matrix for all the sampled points, we arrive at the familiar linear equations for eigendecomposition:

$$\tilde{W}\hat{\Phi} = m\hat{\Lambda}\hat{\Phi}$$

Where $\hat{\Phi} = [\hat{\phi}_1, \ldots, \hat{\phi}_m]$ and $\hat{\Lambda} = diag(\hat{\lambda}_1, \ldots, \hat{\lambda}_m)$. This smaller scale problem can be solved exactly. Approximating the j-th eigenfunction at unsampled point x can be performed with the following:

$$\hat{\phi}_j(x) \approx \frac{1}{m\hat{\lambda}_j}\sum_{i=1}^{m} W(x,y_i)\hat{\phi}_j(y_i)$$

In essence, the Nystrom method as presented in this case reduces the rank of the matrix that needs to be eigendecomposed and approximates eigenfunctions evaluated at unsampled points. For further reference on the Nystrom method [42, 79] and the references therein are useful.

4.4.2 Spectral Clustering via the Nystrom Method

Although the empirical results of spectral clustering is impressive, the storage of the adjacency matrix W grows very quickly for large images. Suppose the image contains P pixels,

the matrix $W \in \mathbb{R}^{P \times P}$. Exact spectral clustering as described earlier in this section, involves memory cost that is prohibitive. In this section we describe a low-rank approximation to W that can be used to perform spectral clustering [17, 38]. For an introduction to the Nystrom method, please refer to **Section 4.4.1**. Suppose we sample $n \ll P$ pixels and partition the adjacency matrix as:

$$W = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}$$

Where $A \in \mathbb{R}^{n \times n}$ is the adjacency matrix computed from the n sampled pixels, $B \in \mathbb{R}^{n \times (P-n)}$ consists of the adjacency computed from the n sampled points with respect to the $P-n$ non-sampled points, and $C \in \mathbb{R}^{(P-n) \times (P-n)}$ is the adjacency between the non-sampled points. Under this formulation, the approximate eigenvectors $\hat{U}$ for W via the Nystrom method takes the following form:

$$\hat{U} = \begin{bmatrix} U \\ B^T U \Lambda^{-1} \end{bmatrix}$$

Where U contains orthogonal eigenvectors of A and Λ is diagonal containing eigenvalues of A. Therefore, $A = U \Lambda U^T$. The low rank representation/approximation of W takes the following form:

$$\begin{aligned} \hat{W} &= \hat{U} \Lambda \hat{U}^T \\ &= \begin{bmatrix} A \\ B^T \end{bmatrix} A^{-1} \begin{bmatrix} A & B \end{bmatrix} \end{aligned}$$

From the formulation above, the low-rank representation of W via Nystrom method approximates the matrix $C \approx B^T A^{-1} B$. Suppose we define $A^{-1/2}$ as the symmetric positive definite square root of the matrix A, $Z = A + A^{-1/2} B B^T A^{-1/2}$, and $Z \in \mathbb{R}^{n \times n}$ is diagonalized as $U_Z \Lambda_Z U_Z^T$. The matrix V defined as:

$$V = \begin{bmatrix} A \\ B^T \end{bmatrix} A^{-1/2} U_Z \Lambda_Z^{-1/2}$$

will be the the matrix of eigenvectors which will diagonalize $\hat{W} = V \Lambda_Z V^T$. Suppose we wish to perform spectral clustering that solves the regularized optimization of the form

(B.1), we need the row sums of $\hat{W}$ which can be computed as

$$\begin{aligned}\mathbf{d} &= \hat{W}\mathbf{1} \\ &= \begin{bmatrix} A\mathbf{1}_n + B\mathbf{1}_{(P-n)} \\ B^T\mathbf{1}_n + B^T A^{-1} B\mathbf{1}_{(P-n)} \end{bmatrix}\end{aligned}$$

We then "normalize" the matrix by replacing the entries of A and B with:

$$\begin{aligned}\tilde{A}_{ij} &\leftarrow \frac{A_{ij}}{\sqrt{d_i d_j}} && \forall i, j = 1, \ldots n \\ \tilde{B}_{ij} &\leftarrow \frac{B_{ij}}{\sqrt{d_i d_j}} && \forall i = 1, \ldots n, j = n+1, \ldots, P\end{aligned}$$

After renormalization we may use the renormalized version of the eigenvectors:

$$\tilde{V} = \begin{bmatrix} \tilde{A} \\ \tilde{B}^T \end{bmatrix} \tilde{A}^{-1/2} U_{\tilde{Z}} \Lambda_{\tilde{Z}}^{-1/2}$$

Where $\tilde{Z} = \tilde{A} + \tilde{Z}^{-1/2}\tilde{B}\tilde{B}^T\tilde{A}^{-1/2}$. The number of groups $\tilde{G}$ is determined by examining the leading eigenvalues given by the diagonal values of $\Lambda_{\tilde{S}}$. Suppose once more we have eigenvalues arranged in descending order $\{\tilde{e}_{(n)}, \ldots \tilde{e}_{(1)}\}$, the number of groups is:

$$\tilde{G} = \max\{n - i : \tilde{e}_{(i)} \geq \tilde{\tau}, i = 1, \ldots n\} \tag{4.7}$$

Note that the low-rank approximation only requires the storage of matrix $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times (P-n)}$ and the diagonalization of the matrix $S \in \mathbb{R}^{n \times n}$ which is significantly less complex than the storage and the diagonalization of the matrix $W \in \mathbb{R}^{P \times P}$. Similar to the group number determination in exact spectral clustering, $\tilde{\tau}$ is dependent on the observed data and setting it requires a judgmental call regarding the number of groups relative to the number of pixels in the image. Once $\tilde{G}$ has been established, we collect $\tilde{G}$ eigenvectors associated with the $\tilde{G}$ largest eigenvalues into matrix $\tilde{E} \in \mathbb{R}^{P \times \tilde{G}}$. We then apply K-means clustering on the P $\tilde{G}$-dimensional row-vectors.

However, the Nystrom method is not without its drawbacks. It is less accurate in terms of its classification. In order to demonstrate this, we conducted an experiment with synthetic data containing 600×600 pixels (which is comparable to the number of pixels in the image from the Gulf Wetlands) with 7 groups. For each of the group, we generated a random β with number of non-zeros set at 15% of its entries. A library X is chosen from the group of libraries extracted from the Reno scene for each group. In addition, each li-

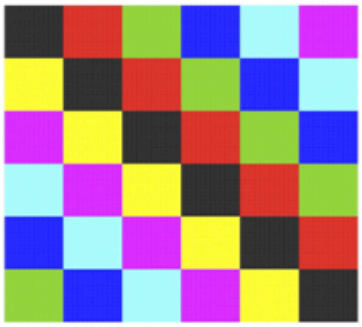
brary is augmented with bilinear combinations resulting in each library having 20 columns (ie. $X \in \mathbb{R}^{356\times 20}$). Each pixel in a given group will have the following observed spectral signature:

$$Y_i = X\beta + \epsilon_i \qquad\qquad \epsilon_i \sim N(0, 0.125^2 I_{356})$$

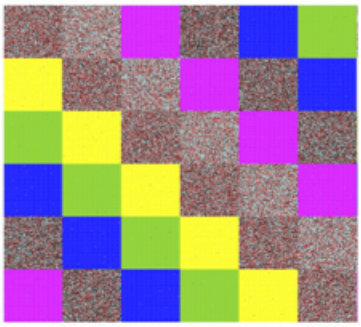
In our experiments we tested the low-rank approximation with different number of sampled pixels used. We experimented with $\tilde{n} \in \{100, 200, 300, 400, 500, 600, 700, 800\}$ sampled pixels and **Table 4.1** contains the average misclassification rate for these experiments over 20 repetitions. As reference, we included classification plots for some interesting cases of the recovery using Nystrom method in **Figure 4.3**. It is important to note that increasing the number of pixels sampled increases the computational burden in terms of memory and operations by the order of $\mathcal{O}(\tilde{n}^2)$ while results of the experiments with synthetic data does not show dramatic improvement in the classification rate. Based on these results, we chose $n = 300$ pixels in our application of the proposed method in real data simulations in order to strike a balance between accuracy and computational complexity.

# sampled	100	200	300	400	500	600	700	800
Error rate	0.2568	0.2773	0.2567	0.2209	0.2464	0.2256	0.2382	0.2081

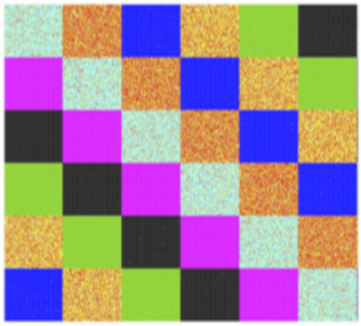
Table 4.1: Misclassification rates using the low-rank approximation to spectral clustering.



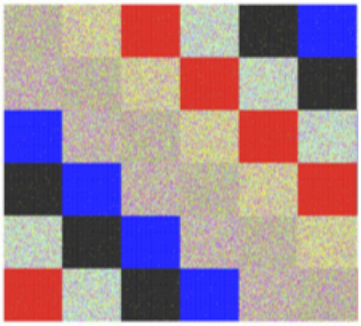
((a)) The ground truth used to generate synthetic data



((b)) Recovery using 300 sampled pixels



((c)) Recovery using 600 sampled pixels



((d)) Recovery using 700 sampled pixels

Figure 4.3: Classification for the ground truth and the recovered classification for different sampled pixels.

4.5 Simulation and Results

This section contains the simulation studies conducted using the proposed methodology and the results of the simulation studies. The simulation will start off with a controlled

experiment using synthetic data set which has known parameters Θ and β from which the synthetic spectral signatures are generated. Note that the updates for both parameters Θ and β are implemented in parallel groups [59] using R and Rcpp [36].

In addition to synthetic data simulation, we also performed the spectral clustering and unmixing on real data sets. The real data sets we used in the following sections are reflectance data obtained from remote sensors and are publicly available at the website listed in the reference section [2]. It is also important to note that the reflectance data is transformed from radiance data detected by the remote sensor via the radiance to reflectance equation [1].

In the real data experiments, we generated the library X using VCA [66]. VCA is an iterative algorithm that extracts endmembers from observed spectral signature in the image. This method assumes that there is at least one pure pixel in the image (pixel group). The algorithm takes the following form:

1. Begin by select a random pixel from the image as an endmember in the library. The first endmember can be taken from the pixel with the largest spectral signature.

2. Find the space that is orthogonal to the column space of the library.

3. From the remaining pixels, the pixel with the largest orthogonal projection is added into the library as an endmember.

4. Repeat steps 2 and 3 until an appropriate number of endmembers are included in the library. It is obvious that the maximum number of endmembers cannot exceed the number of pixels in the image.

From our observation of the recovered endmembers from VCA, only the first few iterations of VCA are needed. We augmented the library with bilinear combinations of the original spectral signatures.

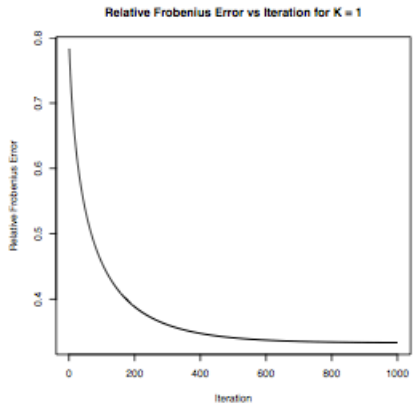
4.5.1 Synthetic Data

We document the synthetic and real data simulation results in this section. The synthetic data involves a 50×50 image consisting of 4 classes. Spectral clustering does a good job of separating the pixels correctly as evidenced by **Figure 4.2**. Within each class, we implemented the algorithm described in **Section 4.3.3** in order to recover the abundances and the structure of the wavelength dependence. We measure the accuracy of the recovered concentration matrix Θ within each group using the relative Frobenius error which is defined

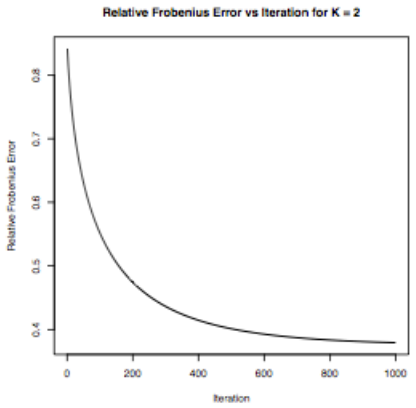
as

$$\text{RelFrob}_{\Theta}(\hat{\Theta}) = \frac{\|\Theta - \hat{\Theta}\|^2}{\|\Theta\|^2}$$

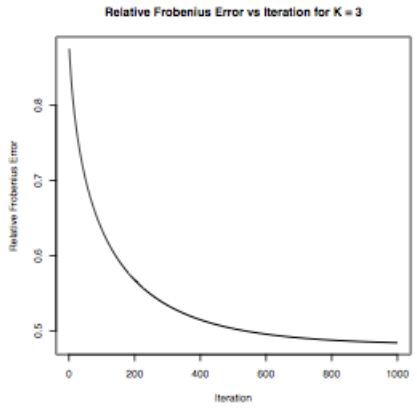
Where Θ is the concentration matrix used to generate the synthetic wavelength dependence in the synthetic data and $\hat{\Theta}$ is the recovered concentration matrix. **Figures 4.4(a) - 4.4(d)** shows how the relative Frobenius error of the concentration matrix decreases as a function of the number of iterations. This shows that iterating the updates Θ_k from (4.6) are converging. The relative l_2 error for the abundances are negligible for all 4 groups as evidenced by **Figures 4.5(a) - 4.5(d)**.



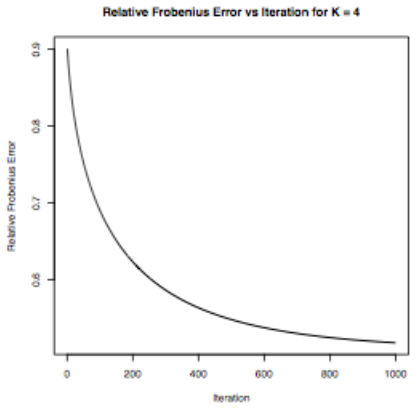
((a)) Relative Frobenius error for group 1



((b)) Relative Frobenius error for group 2

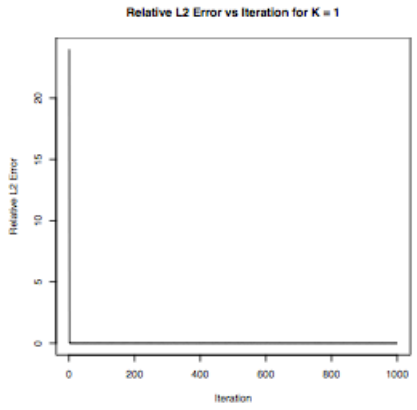


((c)) Relative Frobenius error for group 3

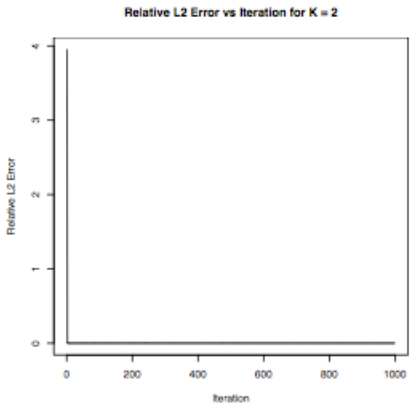


((d)) Relative Frobenius error for group 4

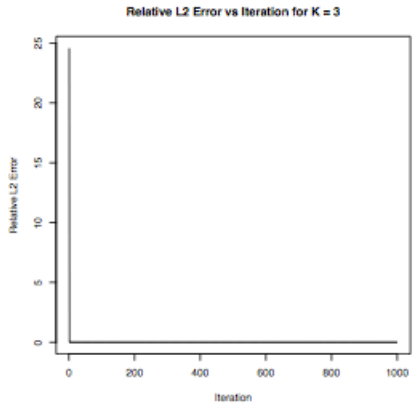
Figure 4.4: Relative Frobenius error of the estimated wavelength dependence as a function of iterations. This provides empirical proof that the algorithm produces a sequence of Θs that converges.



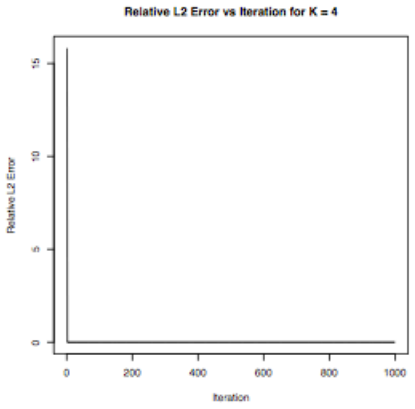
((a)) Relative l_2 error for group 1



((b)) Relative l_2 error for group 2



((c)) Relative l_2 error for group 3



((d)) Relative l_2 error for group 4

Figure 4.5: Relative l_2 error of the estimated abundances as a function of iterations. This provides empirical proof that the algorithm produces a sequence of βs that converges.

4.5.2 Reno Scene

We also applied the methodology described in this chapter on a scene from Reno obtained from [2]. The scene is an urban area from Reno, Nevada contains 600×320 pixels. The scene contains buildings, roads, parking lots, and a river. This scene was chosen because of the distinctive features of the urban environment enables some form of "eyeball" validation of the classification plots using images taken on the visible spectrum. We performed exact spectral clustering on the 100×100 subset of the scene in the exploratory analysis. The classification recovered from the exact spectral clustering can be seen in **Figure 4.6**. We also performed approximate spectral clustering via the Nystrom method as outlined in **Section 4.4.2** on the whole image.

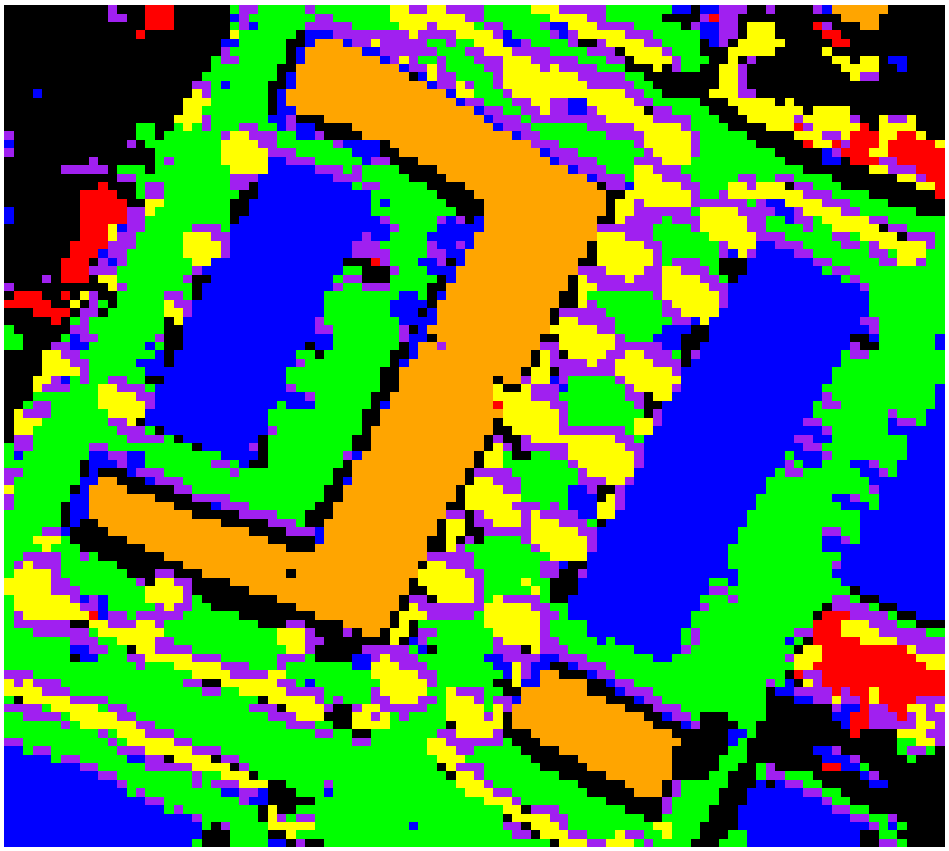
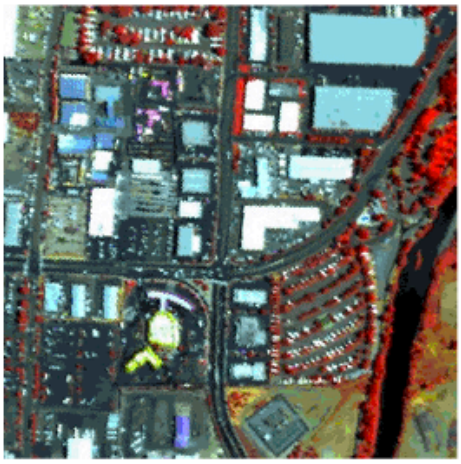


Figure 4.6: Classification plot of a subset of the Reno data using exact spectral clustering.

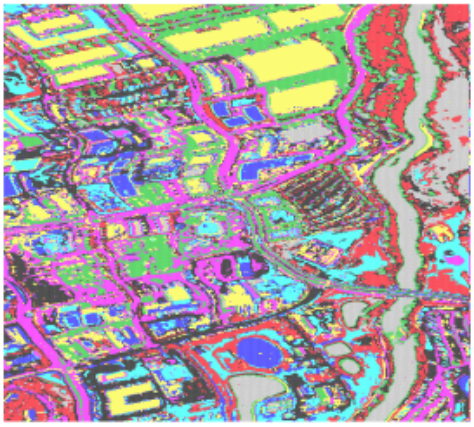
In the approximation, we sampled $n = 300$ pixels from 600×320 pixels and performed spectral clustering utilizing a low-rank approximation of W. In this instance, by examining the eigenvalues $\Lambda_{\tilde{Z}}$ of the diagonalization of $\tilde{Z} \in \mathbb{R}^{300\times300}$, we determined that there are $G = 14$ distinct groups in the image. The K-means clustering is performed on the row vectors of $\tilde{E} \in \mathbb{R}^{(600\times320)\times14}$ which contains the 14 eigenvectors associated with the 14 largest eigenvalues of $\tilde{Z}$. **Figure 4.7** shows the classification plot relative to the image on the visible spectrum.

Pay special attention to the upper left hand corner of **Figure 4.7(b)** and compare it to **Figure 4.6**. From these two classification plots, we can see that there is very little loss

in fidelity in utilizing the low-rank approximation of the adjacency W to perform spectral clustering because the classification plot for the approximate spectral clustering is still able to capture the building and the parking lot covered in the classification plot recovered from exact spectral clustering. This exercise confirms that the loss in fidelity in utilizing a low-rank representation of W for spectral clustering is minimal. Also note that there appears to be attenuation based on the "wavy" appearance of the roads and rivers in the image. This may be an artifact of image processing which converts the radiance data from the remote sensor into reflectance data via the radiance to reflectance equation in [1].



((a)) Visible spectrum photo



((b)) Recovered classification plot

Figure 4.7: Complete classification of the Reno scene using Nystrom method.

After classification, we performed VCA within each class to recover endmembers for the library. In this exercise, we extracted 5 endmembers from the each group of pixels and augmented the 5 endmembers with bilinear combinations resulting in a library containing 20 endmembers (columns). Once we have the library X, we applied the updates outlined in **Sections 4.3.1 - 4.3.2** .

In order to achieve $\frac{10}{L\times L}$ level of sparsity for Θ, we can set $\alpha \approx 0.9$ with $\lambda \propto \sqrt{\frac{L}{|G_i|}}$ where $|G_i|$ is the number of pixels within group G_i. For the Reno scene we set $\alpha = 0.5$ and $\lambda = c\sqrt{\frac{L}{|G_i|}}$. The constant $c \in \{0, 0.025, \ldots, 0.25\}$ is found by grid search evaluating a model selection criterion.

Selecting the penalization level requires a metric that measures the fit of the model while penalizing for model complexity. For most cases, the Bayesian Information Criterion (BIC) as defined below is adequate

$$BIC_\lambda(\hat{\Theta},\hat{\beta}) = -2\log L(\hat{\Theta},\hat{\beta}) + R|\hat{\Theta}|_0 \log|G_i|$$

Where $|G_i|$ is the number of pixels in group i. However, in the estimation of the concentration matrix it is often that we run into cases where the number of parameters (number of non-zeros in the concentration matrix) is growing with the sample size which violates one of the assumption required for BIC consistency [39]. In lieu of the regular BIC as a criterion for selecting c, we used the extended BIC [39] which allows for the growth of the number of non-zero entries in the concentration matrix. The extended BIC takes the following form for group G_i :

$$\tilde{BIC}_\lambda(\hat{\Theta},\hat{\beta}) = -2\log L(\hat{\Theta},\hat{\beta}) + R|\hat{\Theta}|_0 \log|G_i| + 4|\hat{\Theta}|_0 \lambda \log L \tag{4.8}$$

Note the additional penalization term in the extended BIC which means the extended BIC is more punitive towards complex models. The resultant BICs are compared for different values of $c \in \{0, 0.025, \ldots, 0.25\}$ and the one with the lowest value is chosen as the optimal $\lambda^\star$. The estimates are then computed using $\lambda^\star$. The resultant estimates of $\hat{\Theta}$ and $\hat{\beta}$ are taken as the ideal estimates.

In order to gain insight into the improvement spatial dependence and wavelength dependence brings to our model, we performed hyperspectral unmixing on different scenarios which are listed below:

1. Each group has one mixture parameter β without accounting for wavelength dependence. This model accounts for spatial dependence while leaving out wavelength dependence. Hereforth we would refer to this as Model 1.

2. Each pixel in groups have a mixture parameter β without accounting for wavelength dependence. This model is similar to Model 1 in terms of accounting for spatial dependence but has more mixture parameters β which results in a better fit residual-wise. Hereforth we would refer to this as Model 2.

3. Each pixel in the image has a mixture parameter β without accounting for wavelength dependence. This model does not account for spatial and wavelength dependence. Essentially, this model performs pixel level unmixing. The library used in this model is extracted from the whole image rather than at the group level. Hererforth we would refer to this as Model 3.

In order to compare the proposed to the 3 models listed above, we calculated the regular BIC for the 3 models listed and compared them to the extended BIC (eBIC) of the proposed model. The regular BIC for the challenger models:

$$BIC(\hat{\beta}) = -2\log L(\hat{\beta}) + R\log(\text{number of pixels}) \tag{4.9}$$

Note that the extended BIC is just the regular BIC with additional penalties for non-zero values for the concentration matrix. **Table 4.2** documents the BIC/eBIC computed for the models as a comparison. The reason we chose BIC/eBIC as a comparison is to provide a meaningful way to compare model fit for models with different number of parameters. As evidenced from the results, even with the extra penalization term in eBIC, the model with the best fit is the one that incorporates spatial and wavelength dependence.

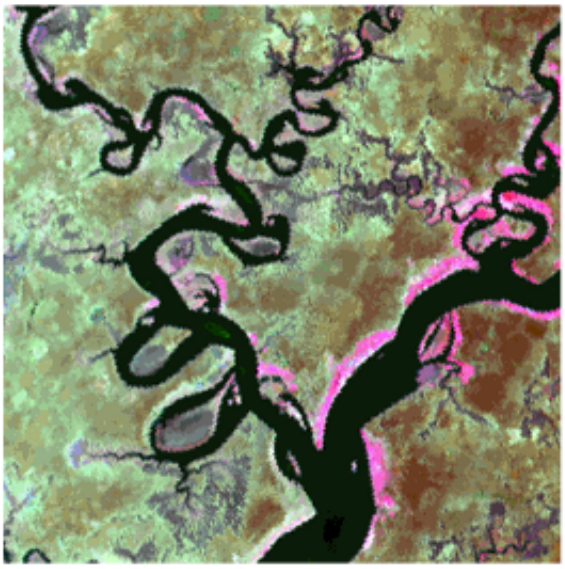
Model	Proposed	Model 1	Model 2	Model 3
BIC/eBIC	81,012,682	125,765,996	165,615,242	179,195,286

Table 4.2: The BIC for the 3 challenger models and extended BIC for the proposed model applied to the Reno scene.

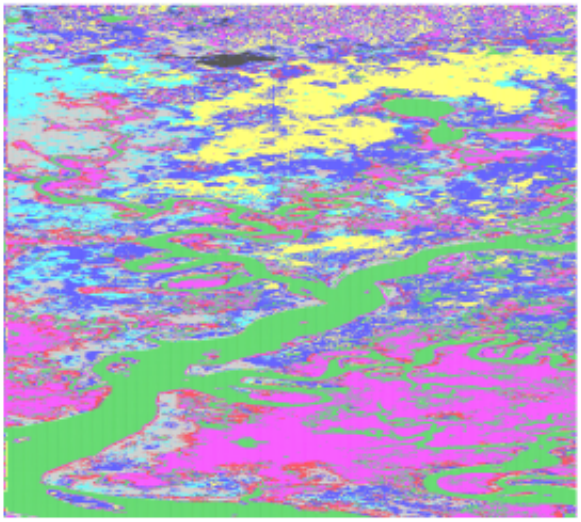
4.5.3 Gulf Wetlands (Suwannee River) Scene

Beside the Reno scene, we also examined a scene from the Suwannee River obtained from [2]. The scene contains a river delta, wetlands, and plants indigenous to swamp lands. The image contains 1200×320 pixels. We performed the approximate spectral clustering using the low-rank approximation of W using 300 sampled pixels and recovered $G = 13$ groups from the scene. **Figure 4.8** contains the classification plot and the image on the visible

spectrum. From the classification plot, we can tell that approximate spectral clustering is able to clearly distinguish the river from the wetlands overgrown with indigenous plants.



((a)) Visible spectrum photo



((b)) Recovered classification plot

Figure 4.8: Complete classification of the Gulf Wetlands (Suwannee River) scene using Nystrom method.

In addition, we also performed similar comparisons via the BIC/eBIC relative to the 3 challenger models as described in **Section 4.5.2**. **Table 4.3** documents the resultant BIC/eBIC computed for the models as a comparison between the models. In this case, the improvement in model fit is more dramatic when compared to the Reno scene.

Model	Proposed	Model 1	Model 2	Model 3
BIC/eBIC	110,079,432	254,101,067	335,344,801	352,905,313

Table 4.3: The BIC for the 3 challenger models and extended BIC for the proposed model applied to the Gulf Wetlands (Seuannee River) scene.

4.6 Conclusion

The method is novel because it incorporates both spatial and wavelength dependence in hyperspectral unmixing and classification. Empirical evidence from the synthetic data simulation shows that the algorithm converges for both the mixing and wavelength correlation parameter. Based on the results shown in the simulation. especially from the Reno and Suwannee scenes, the incorporation of spatial and wavelength correlation improve the model fit. Spectral clustering is able to accurately classify the pixels in the images. However, spectral clustering is computationally (in terms of memory and computational operations) demanding for larger images. However, approximate spectral clustering using low-rank approximation enables the approximate classification of large images. The increase in the number of samples in the approximate spectral clustering does not bring dramatic improvement to the classification so the sample size in the Nystrom approximation has to be balanced between the accuracy of classification and the computational burden involved.

APPENDIX A

Appendix: Simultaneous Unmixing and Classification of Hyperspectral Images

A.1 Optional: Estimation of the Granularity Constant θ

Throughout the write-up of this document, we assumed the granularity constant is fixed. Optimizing with respect to the granularity constant is omitted because the addition of the granularity updates increases the complexity of the algorithm of the EM algorithm significantly without notable gains in either classification and model fit as seen in **Table 3.1** and **Table 3.2**

However, it is possible to estimate the strength of the spatial dependence. It is not possible to solve explicitly for θ. However, it may be possible to implement a gradient algorithm similar to the one outlined in [56] using the first and second derivative of Q_t with respect to θ_t. Once the first and second derivatives of Q_t as evaluated below apply the following update to β at each iteration of the EM Algorithm:

$$\theta_{t+1} = \theta_t - \frac{dQ_t}{d\theta_t}\left[\frac{d^2Q_t}{d\theta_t^2}\right]^{-1} \tag{A.1}$$

Since the rate of convergence is quadratic, only one iteration of this is required for each iteration of the EM algorithm.

Remark 6. *It should be noted that, the gradient update is not exactly the same as [56] due to the fact that the first and second derivatives are not evaluated directly in this case. Both these quantities are calculated using a combination of an algorithm similar to* ***Section 3.4.2*** *(to sample from $f(z|\beta)$) and Monte Carlo (to evaluate both expectations).*

$$
\begin{aligned}
G(\theta_t) &= \sum_{\mathbf{z}\in\{1,\ldots,K\}^P} \exp\left\{\theta_t \sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\} \\
G'(\theta_t) &= \sum_{\mathbf{z}\in\{1,\ldots,K\}^P} \exp\left\{\theta_t \sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\} \sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'}) \\
G''(\theta_t) &= \sum_{\mathbf{z}\in\{1,\ldots,K\}^P} \exp\left\{\theta_t \sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\} \left\{\sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\}^2 \\
\frac{dQ_t}{d\theta_t} &= -\frac{G'(\theta_t)}{G(\theta_t)} + w_t \\
\frac{d^2Q_t}{d\theta_t^2} &= -\frac{G(\theta_t)G''(\theta_t) - G'(\theta_t)^2}{G(\theta_t)^2} \\
&= -\frac{G''(\theta_t)}{G(\theta_t)} + \left[\frac{G'(\theta_t)}{G(\theta_t)}\right]^2
\end{aligned}
$$

Note that the both the first and second derivatives are a form of expectation due to the following:

$$
\begin{aligned}
\frac{G'(\theta_t)}{G(\theta_t)} &= \sum_{\mathbf{z}\in\{1,\ldots,K\}^P} \frac{\exp\left\{\theta_t \sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\}}{G\theta_t)} \sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'}) \\
&= \sum_{\mathbf{z}\in\{1,\ldots,K\}^P} \left\{\sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\} f(\mathbf{z}|\theta_t) \\
&= \mathbb{E}\left[\sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right] \\
\frac{G''(\theta_t)}{G(\theta_t)} &= \sum_{\mathbf{z}\in\{1,\ldots,K\}^P} \frac{\exp\left\{\theta_t \sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\}}{G(\theta_t)} \left\{\sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\}^2 \\
&= \sum_{\mathbf{z}\in\{1,\ldots,K\}^P} \left\{\sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\}^2 f(\mathbf{z}|\theta_t) \\
&= \mathbb{E}\left[\left\{\sum_{p=1}^{P} \sum_{p'\in\mathcal{V}(p)} \delta(z_p - z_{p'})\right\}^2\right]
\end{aligned}
$$

The expectations in the gradient expressions are with respect to the Potts-Markov distribu-

tion. Recall the Potts-Markov distribution is of the following form:

$$f(\mathbf{z}|\theta) = \frac{1}{G(\theta)} \exp\left\{ \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta\delta(z_p - z_{p'}) \right\},$$

Sampling from this distribution is not trivial. The sampling algorithm requires the used of the MCMC algorithm described in **Section 3.4.2** with the likelihood ratio set to $\frac{\pi(\tilde{z})}{\pi(z)} = 1$. It should be noted that the gradient as described is an approximate gradient. Due to the squared convergence rate of the gradient method only one iteration of the gradient is required for each EM iteration according to [56].

A.2 Markov Property of Potts-Markov Model

In this section we document an interesting property of the Potts-Markov model. Suppose we define $\mathcal{V}(i)$ as the first order neighbors of the pixel i as illustrated in Figure A.1 .

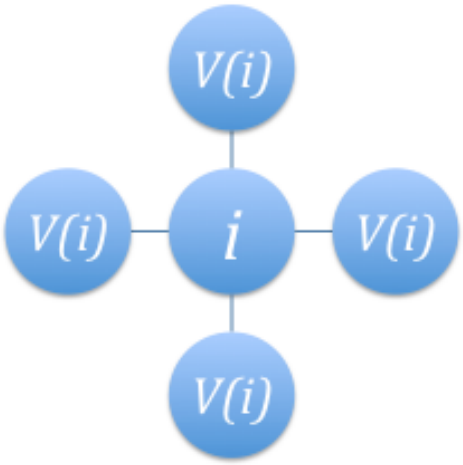


Figure A.1: The first order neighbor of pixel i.

We observe that the Potts-Markov model has Markov property whereby a pixel is only

dependent on its first order neighbors:

$$
\begin{aligned}
f(\mathbf{z}|\theta) &= \frac{1}{G(\theta)} \exp\left\{ \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta \delta(z_p - z_{p'}) \right\} \\
f(z_i|\mathbf{z}_{-i}) &= \frac{f(\mathbf{z}|\theta)}{f(\mathbf{z}_{-i}|\theta)} \\
&= \frac{\frac{1}{G(\theta)} \exp\left\{ \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta \delta(z_p - z_{p'}) \right\}}{\sum_{z_i=1}^{K} \frac{1}{G(\theta)} \exp\left\{ \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta \delta(z_p - z_{p'}) \right\}} \\
&= \frac{\frac{1}{G(\theta)} \exp\left\{ \sum_{p=1}^{P} \sum_{p' \in \mathcal{V}(p)} \theta \delta(z_p - z_{p'}) \right\}}{\frac{1}{G(\theta)} \exp\left\{ \sum_{p \neq i} \sum_{p' \in \mathcal{V}(p)} \theta \delta(z_p - z_{p'}) \right\} \sum_{z_i=1}^{K} \exp\left\{ \sum_{p' \in \mathcal{V}(i)} \theta \delta(z_i - z_{p'}) \right\}} \\
&= \frac{\exp\left\{ \sum_{p' \in \mathcal{V}(i)} \theta \delta(z_i - z_{p'}) \right\}}{\sum_{z_i=1}^{K} \exp\left\{ \sum_{p' \in \mathcal{V}(i)} \theta \delta(z_i - z_{p'}) \right\}} \\
&= f(z_i|\mathbf{z}_{\mathcal{V}(i)})
\end{aligned}
$$

APPENDIX B

Appendix: Hyperspectral Unmixing with Wavelength Dependence

B.1 Conditional Independence of Multivariate Normal

For $X \in \mathbb{R}^p$ and $X \sim N(0, \Sigma)$. Defining the concentration matrix as $\Theta = \Sigma^{-1}$, prove that X_i is conditionally independent of X_j given X_{-ij} if $\theta_{ij} = \theta_{ji} = 0$.

Proof. Without loss of generality, suppose $\sigma_{12} = \sigma_{21} = 0$. Then partition the concentration:

$$\Theta = \left(\begin{array}{c|c} \Theta_{12} & M \\ \hline M^T & \Theta_{-12} \end{array}\right)$$

$$\Theta_{-12} = \begin{pmatrix} \theta_{33} & \theta_{34} & \cdots & \theta_{3p} \\ \theta_{43} & \theta_{44} & \cdots & \theta_{4p} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{p3} & \theta_{p4} & \cdots & \theta_{pp} \end{pmatrix} \qquad M = \begin{pmatrix} \theta_{13} & \theta_{14} & \cdots & \theta_{1p} \\ \theta_{23} & \theta_{24} & \cdots & \theta_{2p} \end{pmatrix}$$

$$\Theta_{12} = \begin{pmatrix} \theta_{11} & 0 \\ 0 & \theta_{22} \end{pmatrix} \qquad X^T = \begin{pmatrix} X_{12}^T X_{-12}^T \end{pmatrix}$$

Therefore, the square term in the exponent of the multivariate normal is:

$$X^T \Theta X = X_{12}^T \Theta_{12} X_{12} + X^T \begin{pmatrix} 0 & M \\ M^T & 0 \end{pmatrix} X + X_{-12}^T \Theta_{-12} X_{-12}$$

$$\begin{aligned} X^T \begin{pmatrix} 0 & M \\ M^T & 0 \end{pmatrix} X &= 2 X_{12}^T M X \\ &= 2X_1 \sum_{j=1}^{p} X_j \theta_{1j} + 2X_2 \sum_{j=1}^{p} X_j \theta_{2j} \end{aligned}$$

Note:

$$f(X_{-12}) \propto \exp\left\{-\frac{1}{2}X_{-12}^T\Theta_{-12}X_{-12}\right\}$$

$$\therefore f(X_{12}|X_{-12}) = \frac{f(X)}{f(X_{-12})}$$

$$\propto \exp\left\{-\frac{1}{2}\left[X_{12}^T\Theta_{-12}X_{12} + 2X_{12}^TMX\right]\right\}$$

Since there are no X_1X_2 terms in the exponent(separable), $f(X_{12}|X_{-12}) = f(X_1|X_{-12})\dot{f}(X_2|X_{-12})$. Therefore, X_1 is conditionally independent of X_2 given X_{-12} if $\theta_{12} = \theta_{21} = 0$. For arbitrary X_i and X_j, just rearrange the variables (assuming the concentration matrix is permutation invariant), and proceed accordingly. □

B.2 Intuition for Spectral Clustering

Spectral clustering is an empirical approach to cluster pixels using eigenvectors of matrices derived from data associated with the pixels [70]. The method involves the use of an adjacency/weight matrix constructed via a distance kernel applied to the pixels. Intuitively, the method seeks to group pixels in such a way that the weights between pixels belonging to the same group (intra-group weights) are large while the weights between pixels belonging to different groups (inter-group weights) are small. This means the pixels within the same group are more similar to each other while the pixels belonging to different groups are dissimilar to each other. Suppose we define the the weights between pixels in group A and B as $W(A,B) = \sum_{i\in A, j\in B} W_{ij}$ and A^C is the complement of group A, [58] states that the spectral clustering problem is the choice of partition $A_1,\ldots,A_K$ such that the following is minimized:

$$\hat{A}_1,\ldots,\hat{A}_G = \arg\min_{A_1,\ldots,A_G} \frac{1}{2}\sum_{i=1}^{K} W(A_i, A_i^C)$$

The fraction $\frac{1}{2}$ is added because the adjacency matrix as outlined above is symmetric. In practice, the solution to this optimization problem does not lead to a satisfactory partition because it usually leads to groups that contain only one pixel. Therefore, [83] suggested normalized partitions of the form:

$$\hat{A}_1,\ldots,\hat{A}_G = \arg\min_{A_1,\ldots,A_G} \frac{1}{2}\sum_{i=1}^{G} \frac{W(A_i, A_i^C)}{vol(A_i)} \tag{B.1}$$

Where $vol(A_i)$ is the sum of the weights within group A_i. Alternatively, [46] proposed the following:

$$\hat{A}_1, \ldots, \hat{A}_G = \arg \min_{A_1,\ldots,A_G} \frac{1}{2} \sum_{i=1}^{G} \frac{W(A_i, A_i^C)}{|A_i|}$$

Where $|A_i|$ is the number of pixels within group A_i. Both contain regularizations that encourage each partition A_i to be "reasonably large".

BIBLIOGRAPHY

[1] SpecTIR Website. https://eo1.usgs.gov/faq/question?id=21.

[2] USGS Website. http://www.spectir.com/free-data-samples/.

[3] R. Ablin and C. Helen Sulochana. A survey of hyperspectral image classification in remote sensing. *International Journal of Advanced Research in Computer and Communication Engineering*, 2:2986–3000, 2013.

[4] Elizaveta Levina Ji Zhu Adam J. Rothman, Peter J. Bickel. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, pages 494–515, 2008.

[5] M. Afonso, J. Bioucas-Dias, and M. A. T. Figueiredo. A fast algorithm for the constrained formulation of compressive image reconstruction and other linear inverse problems. In *IEEE International Conf. on Acoustics, Speech, and Signal Processing - ICASSP*, pages –, March 2010.

[6] Y Altmann, M Pereyra, and J. Bioucas-Dias. Collaborative sparse regression using spatially correlated supports - application to hyperspectral unmixing. *IEEE Trans. on Image Processing*, 24(12):5800–5811, December 2015.

[7] Rita Ammanouil, André Ferrari, and Cédric Richard. A graph laplacian regularization for hyperspectral data unmixing. *CoRR*, abs/1410.3699, 2014.

[8] Yves F. Atchadé. A computational framework for empirical bayes inference. *Statistics and Computing*, 21(4):463–473, 2011.

[9] Yves F. Atchade, Rahul Mazumder, and Jie Chen. Scalable computation of regularized precision matrices via stochastic optimization, 2015.

[10] S. Balakrishnan and D. Madigan. Priors on the variance in sparse bayesian learning: the demi-bayesian lasso. In M-H Chen, P. Muller, D. Sun, and K. Ye, editors, *Frontiers of Statistical Decision Making and Bayesian Analysis: In Honor of James O. Berger*. Springer, New York, 2010.

[11] A.m. Baldridge, S.j. Hook, C.i. Grove, and G. Rivera. The aster spectral library version 2.0. *Remote Sensing of Environment*, 113(4):711715, 2009.

[12] Dimitri P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods (Optimization and Neural Computation Series)*. Athena Scientific, 1 edition, 1996.

[13] Peter J. Bickel, Ya'acov Ritov, and Alexandre B. Tsybakov. Simultaneous analysis of lasso and Dantzig selector. *Ann. Statist.*, 37(4):1705–1732, 2009.

[14] J. Bioucas-Dias and M. A. T. Figueiredo. Alternating direction algorithms for constrained sparse regression: Application to hyperspectral unmixing. In *IEEE GRSS Workshop on Hyperspectral Image and Signal Processing*, volume 1, pages –, June 2010.

[15] Joseph W. Boardman. Automating Spectral Unmixing of AVIRIS Data Using Convex Geometry Concepts. *Proc. Ann. JPL Airborne Geosci. Workshop*, 1:11–14, 1993.

[16] Leonard Bottolo and Sylvia Richardson. Evolutionary stochastic search for bayesian model exploration. *Bayesian Anal.*, 5(3):583–618, 09 2010.

[17] Djallel Bouneffouf and Inanc Birol. Sampling with minimum sum of squared similarities for nyström-based large scale spectral clustering. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 2313–2319. AAAI Press, 2015.

[18] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, January 2011.

[19] Joshua Broadwater and Amit Banerjee. Mapping intimate mixtures using an adaptive kernel-based technique. *2011 3rd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 2011.

[20] Peter Bühlmann and Sara van de Geer. *Statistics for high-dimensional data*. Springer Series in Statistics. Springer, Heidelberg, 2011. Methods, theory and applications.

[21] G. Camps-Valls, D. Tuia, L. Bruzzone, and J. Atli Benediktsson. Advances in hyperspectral image classification: Earth monitoring with statistical learning methods. *Signal Processing Magazine, IEEE*, 31(1):45–54, Jan 2014.

[22] Emmanuel J. Cands. Modern statistical estimation via oracle inequalities. *ANU Acta Numerica*, 15:257, 2006.

[23] O. Cappe, Moulines E., and T. Ryden. *Inference in hidden Markov models*. Springer series in Statistics. New York, 2005.

[24] B. P. Carlin and A. E. Gelfand. Approaches for empirical Bayes confidence intervals. *JASA*, 85(409):105–114, 1990.

[25] B. P. Carlin and T. A. Louis. Empirical Bayes: Past, Present and Future. *JASA*, 95(452):1286–1289, 2000.

[26] Carlos M. Carvalho, Nicholas G. Polson, and James G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

[27] G. Casella. Empirical Bayes Gibbs sampling. *Biostatistics*, 2:485–500,, 2001.

[28] G Casella and E I George. Explaining the Gibbs sampler. *The American Statistician*, 46:167–174, 1992.

[29] J. Chen, C. Richard, and P. Honeine. Nonlinear unmixing of hyperspectral data based on a linear-mixture/nonlinear-fluctuation model. *IEEE Transactions on Signal Processing*, 61(2):480–492, Jan 2013.

[30] R.N. Clark, G.A. Swayze, K.B. Heiderbrecht, R.O. Green, and A.F.H. Goetz. Calibration to surface reflectance of terrestrial imaging spectrometry data. *Summaries of the 5th Annual JPL airborne earth science workshop*, page 4142, 1995.

[31] G.B. Dantzig. *Linear Programming and Extensions*. Princeton landmarks in mathematics and physics. Princeton University Press, 1963.

[32] George B. Dantzig and Philip Wolfe. Decomposition principle for linear programs. *Oper. Res.*, 8(1):101–111, February 1960.

[33] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm (with discussion). *Journal of the Royal Statistical Society. Series B*, 39:1–38, 1977.

[34] Nicolas Dobigeon, Jean-Yves Tourneret, Cedric Richard, Jose Carlos M. Bermudez, Stephen McLaughlin, and Alfred Hero. Nonlinear unmixing of hyperspectral images. *IEEE Signal Processing Magazine*, pages 82–94, 2013.

[35] Jonathan Eckstein and Dimitri P. Bertsekas. On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Program.*, 55(3):293–318, June 1992.

[36] Dirk Eddelbuettel. *Seamless R and C++ Integration with Rcpp*. Springer, New York, 2013. ISBN 978-1-4614-6867-7.

[37] A. C. Faul and M. E. Tipping. Analysis of sparse bayesian learning. *Advances in Neural Information Processing Systems*, 14:383–389, 2002.

[38] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the nyström method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):214–225, January 2004.

[39] Rina Foygel and Mathias Drton. Extended bayesian information criteria for gaussian graphical models. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 604–612. Curran Associates, Inc., 2010.

[40] Jerome H. Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2 2010.

[41] E. I. George and D. P. Foster. Calibration and empirical Bayes variable selection. *JASA*, 87(4):731–747, 2000.

[42] Michael A. Golberg. *Numerical solution of integral equations*. Plenum Press, 1990.

[43] G. Golub and C. F. Van Loan. *Matrix Computations, 4th Ed.* John Hopkins University Press. Baltimore, 2013.

[44] Peter J. Green and Sylvia Richardson. Hidden markov models and disease mapping. *Journal of the American Statistical Association*, 97:1055–1070, 2001.

[45] M. G. Gu and F. H. Kong. A stochastic approximation algorithm with Markov Chain Monte Carlo method for incomplete data estimation problems. *Proc. Natl. Acad. Sci. USA*, 95:7270–7274, 1998.

[46] L. Hagen and A. B. Kahng. New spectral methods for ratio cut partitioning and clustering. *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, 11(9):1074–1085, November 2006.

[47] Abderrahim Halimi, Yoann Altmann, Nicolas Dobigeon, and Jean-Yves Tourneret. Nonlinear unmixing of hyperspectral images using a generalized bilinear model. *IEEE Trans. Geoscience and Remote Sensing*, 49(11):4153–4162, 2011.

[48] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5):303–320, 1969.

[49] M. Iordache, J. Bioucas-Dias, and A. Plaza. Hyperspectral unmixing with sparse group lasso (accepted). In *IEEE International Geoscience and Remote Sensing Symp.- IGARSS*, volume 1, pages 1–4, July 2011.

[50] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza. Sparse unmixing of hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 49(6):2014–2039, June 2011.

[51] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza. Total variation spatial regularization for sparse hyperspectral unmixing. *IEEE Transactions on Geoscience and Remote Sensing*, 50(11):4484–4502, Nov 2012.

[52] Hemant Ishwaran and J. Sunil Rao. Spike and slab variable selection: frequentist and Bayesian strategies. *Ann. Statist.*, 33(2):730–773, 2005.

[53] Hemant Ishwaran and J. Sunil Rao. Spike and slab variable selection: Frequentist and bayesian strategies. *Ann. Statist.*, 33(2):730–773, 04 2005.

[54] Robert Tibshirani Jerome Friedman, Trevor Hastie. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, pages 432–441, 2008.

[55] N. M. Laird and T. A. Louis. Empirical Bayes confidence intervals based on Bootstrap samples. *JASA*, 82(399):739–750, 1987. (with discussion).

[56] Kenneth Lange. A gradient algorithm locally equivalent to the em algorithm. *Journal of the Royal Statistical Society*, pages 425–437, 1995.

[57] Guolan Lu and Baowei Fei. Medical hyperspectral imaging: a review. *J. Biomed. Opt Journal of Biomedical Optics*, 19(1):010901, 2014.

[58] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, December 2007.

[59] Q. Ethan. McCallum and Stephen Weston. *Parallel R*. O'Reilly Media, 2012.

[60] N. Meinhausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. *Ann. Statist.*, 37(1):246–270, 2009.

[61] Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society, Series B*, 72:417–473, 2010.

[62] C. N. Morris. Parametric empirical Bayes inference: Theory and applications. *JASA*, 78:47–65, 1983.

[63] G. Moser and S. B. Serpico. Combining Support Vector Machines and Markov Random Fields in an Integrated Framework for Contextual Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(5):2734–2752, May 2013.

[64] G. Moser, S.B. Serpico, and J.A. Benediktsson. Land-cover mapping by markov modeling of spatial-contextual information in very-high-resolution remote sensing images. *Proceedings of the IEEE*, 101(3):631–651, March 2013.

[65] J. Nascimento and J. Bioucas-Dias. Unmixing hyperspectral intimate mixtures. In *SPIE - Conf. on Image and Signal Processing for Remote Sensing*, volume 7830, pages 78300C–78300C–8, September 2010.

[66] J. M. P. Nascimento and J.M. Bioucas-Dias. Vertex Component Analysis: A Fast Algorithm to Unmix Hyperspectral Data. *IEEE Trans. on Geosci and Remote Sens.*, 43:898–910, 2005.

[67] Jose M. P. Nascimento and Jose M. Bioucas-Dias. Nonlinear Mixture Model for Hyperspectral Unmixing. *Proc. SPIE*, 7477:74770I–74770I–8, 2009.

[68] Sahand N. Negahban, Pradeep Ravikumar, Martin J. Wainwright, and Bin Yu. A unified framework for high-dimensional analysis of *m*-estimators with decomposable regularizers. *Statistical Science*, 27(4):538–557, 11 2012.

[69] R. A. Neville, K. Staenz, T. Szeredi, J. Lefebvre, and P. Hauff. Automatic Endmember Extraction from Hyperspectral Data for Mineral Exploration. *Proc. Canadian Symp. Remote Sens.*, pages 21–24, 1999.

[70] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, 2002.

[71] R. B. O'Hara and M. J. Sillanpää. A review of Bayesian variable selection methods: what, how and which. *Bayesian Anal.*, 4(1):85–117, 2009.

[72] Neal Parikh and Stephen Boyd. Proximal algorithms. *Found. Trends Optim.*, 1(3):127–239, January 2014.

[73] Trevor Park and George Casella. The Bayesian lasso. *J. Amer. Statist. Assoc.*, 103(482):681–686, 2008.

[74] Jie Peng, Pei Wang, Nengfeng Zhou, and Ji Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486):735746, 2009.

[75] Elizaveta Levina Peter J. Bickel. Regularized estimation of large covariance matrices. *The Annals of Statistics*, pages 199–227, 2008.

[76] Antonio Plaza and Chein-I Chang. An improved n-findr algorithm in implementation, 2005.

[77] Y. Qian, S. Jia, J. Zhou, and A. Robles-Kelly. Hyperspectral Unmixing via L1/2 sparsity-constrained nonnegative matrix factorization. *IEEE Transactions on Geoscience and Remote Sensing*, 49(11):4282–4297, Nov 2011.

[78] Pradeep Ravikumar, Martin J. Wainwright, Garvesh Raskutti, and Bin Yu. High-dimensional covariance estimation by minimizing ?lt;subgt;1lt;/subgt;-penalized log-determinant divergence. *Electron. J. Statist.*, 5:935–980, 2011.

[79] H. J. Reinhardt. *Analysis of approximation methods for differential and integral equations*. Springer-Verlag, 1985.

[80] Y. E. Salehani, S. Gazor, S. Yousefi, and I. M. Kim. Adaptive lasso hyperspectral unmixing using admm. In *Communications (QBSC), 2014 27th Biennial Symposium on*, pages 159–163, June 2014.

[81] Daniel V. Samarov, Jeeseong Hwang, and Maritoni Litorja. The spatial lasso with applications to unmixing hyperspectral biomedical images. *Technometrics*, 57(4):503513, Feb 2015.

[82] Robert A. Schowengerdt. *Remote Sensing, Third Edition: Models and Methods for Image Processing*. Academic Press, Inc., Orlando, FL, USA, 2006.

[83] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, August 2000.

[84] Tingni Sun and Cun-Hui Zhang. Scaled sparse linear regression. *Biometrika*, 2012.

[85] Yuliya Tarabalka, Mathieu Fauvel, Jocelyn Chanussot, and Jn Atli Benediktsson. Svm and mrf- based method for accurate classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, pages 640–736, 2010.

[86] Henry Teicher. Exponential bounds for large deviations of sums of unbounded random variables. *Sankhyā Ser. A*, 46(1):41–53, 1984.

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

[88] Michael E. Tipping. Sparse Bayesian Learning and the Relevance Vector Machine. *J. Mach. Learn. Res.*, 1(3):211–244, 2001.

[89] Olivier Eches Nicolas Dobigeon Jean-Yves Tourneret. Enhancing Hyperspectral Image Unmixing with Spatial Correlations with Spatial Correlations. *IEEE Transactions on Geoscience and Remote Sensing*, pages 4239–4247, 2011.

[90] Sara A. van de Geer and Peter Bhlmann. On the conditions used to prove oracle results for the lasso. *Electron. J. Statist.*, 3:1360–1392, 2009.

[91] Aad W. van der Vaart and Jon A. Wellner. *Weak convergence and empirical processes*. Springer Series in Statistics. Springer-Verlag, New York, 1996. With applications to statistics.

[92] R Warren and S Osher. Hyperspectral unmixing by the alternating direction method of multipliers. *Inverse Problems and Imaging*, 14(3):917–933, 2015.

[93] G. C. G. Wei and M. A. Tanner. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704, 1990.

[94] M. West. On scale mixtures of normal distributions. *Biometrika*, 74:446–448, 1987.

[95] Michael E. Winter. N-FINDR: An Algorithm for Fast Autonomous Spectral Endmember Determination in Hyperspectral Data. *Proc. SPIE, Imaging Spectrometry V*, 3753:266–277, 1999.

[96] David P. Wipf and Bhaskar D. Rao. Sparse Bayesian learning for basis selection. *IEEE Trans. Signal Process.*, 52(8):2153–2164, 2004.

[97] D.P. Wipf, B.D. Rao, and S. Nagarajan. Latent variable bayesian models for promoting sparsity. *Information Theory, IEEE Transactions on*, 57(9):6236–6255, Sept 2011.

[98] Ulli Wolff. Collective monte carlo updating for spin systems. *Phys. Rev. Lett.*, 62:361–364, Jan 1989.

[99] Stephen J. Wright. Coordinate descent algorithms. *Math. Program.*, 151(1):3–34, June 2015.

[100] Chia Chye Yee and Yves Atchade. On the Sparse Bayesian Learning of Linear Models. *Communication in Statistics: Publication to Appear*, 2015.

[101] Chia Chye Yee and Yves Atchade. Simultaneous Unmixing and Classification of Hyperspectral Images. *In Progress*, 2016.

[102] A. Zare and P. Gader. Sparsity promoting iterated constrained endmember detection in hyperspectral imagery. *IEEE Geoscience and Remote Sensing Letters*, 4(3):446–450, July 2007.

[103] Peng Zhao and Bin Yu. On model selection consistency of LASSO. *JMLR*, 7:2541–2563, 2006.

[104] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.